Multispectral Image Labeling for Unmanned Ground Vehicle Environments

Michael B. Teresi

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Mechanical Engineering

Alfred L. Wicks, Chair
Kevin B. Kochersberger
Kathleen Meehan

March 20, 2015
Blacksburg, Virginia

Multispectral, multi-spectral, material classification, image labeling, polarimetry, near infrared, short-wave infrared, visible, ground vehicle

Multispectral Image Labeling for Unmanned Ground Vehicle Environments

Michael B. Teresi

Described is the development of a multispectral image labeling system with emphasis on Unmanned Ground Vehicles(UGVs). UGVs operating in unstructured environments face significant problems detecting viable paths when LIDAR is the sole source for perception. Promising advances in computer vision and machine learning has shown that multispectral imagery can be effective at detecting materials in unstructured environments [1][2][3][4][5][6]. This thesis seeks to extend previous work[6][7] by performing pixel level classification with multispectral features and texture. First the images are spatially registered to create a multispectral image cube. Visual, near infrared, shortwave infrared, and visible/near infrared polarimetric data are considered. The aligned images are then used to extract features which are fed to machine learning algorithms. The class list includes common materials present in rural and urban scenes such as vehicles, standing water, various forms of vegetation, and concrete. Experiments are conducted to explore the data requirement for a desired performance and the selection of a hyper-parameter for the textural features. A complete system is demonstrated, progressing from the data collection and labeling to the analysis of the classifier performance.

Acknowledgments

First I would like to express my gratitude to my advisor Dr. Alfred Wicks for his continued support and guidance on my masters research. He has consistently brought the most interesting projects into his lab, and I have benefited greatly from the opportunities he has provided. He has tenaciously challenged us to improve ourselves and treats his students with the utmost respect.

I would also like to thank the remaining members of my committee, Dr. Kevin Kochersberger and Dr. Kathleen Meehan, who have provided critical guidance especially in the early stages of my academic endeavors.

My sincere thanks also goes to Dr. John Bird and Dr. Jonathan Gaines with whom my research began. I have been introduced to many fundamental skills under their tutelage, and I am thankful for the time we worked together. The same goes for my classmates during this period: Ben Goldman and Mark Umansky. In particular I would like to thank Mark for his work on the polarimetric camera used in this thesis. The polarimetric imagery was singular in the features produced and I am thankful for the opportunity to have included it in the analysis.

I would like to thank Peter King, Steve Cacciola, and Ruel Faruque for their collaboration during my years under Dr. Wicks. They have demonstrated the utmost professionalism and generosity when working with my colleagues and I.

Finally, I would like to thank my colleagues in the Mechatronics Lab at Virginia Tech for the lessons I have learned from them during our collaborations as Graduate Research Assistants. James Burns in particular has provided crucial technical assistance in software programming and machine learning. Daniel Moodie likewise has shown me how to approach problems as an engineer and to learn quickly despite an initial lack of knowledge. These collaborations with my colleagues have been what I consider to be indispensable to my development as an engineer.

All photos are by author unless otherwise stated.

Contents

List of Figures

List of Tables

List of Acronyms

$CIE\ L^*a^*b^*$	1976 $CIE\ L^*a^*b^*$ Space. 17, 51, 106
$PC_{0,-45}$	Polarization Contrast I_0, I_{-45}. 67
$PC_{0,45}$	Polarization Contrast I_0, I_{45}. 67
$PC_{0,90}$	Polarization Contrast I_0, I_{90}. 67, 68
$PI_{-45°}$	Polarization Camera 3: $-45°$. 64, 86
$PI_{0°}$	Polarization Camera 0: $0°$. 64, 86
$PI_{45°}$	Polarization Camera 2: $45°$. 64, 86
$PI_{90°}$	Polarization Camera 1: $90°$. 64, 86
AUC	Area Under the Curve. 82
AVIRIS	Airbone Visible/ Infared Imaging Spectrometer. 7
C-SVM	C-Support Vector Machine. 22, 71, 77, 96–98, 116, 139, 140
CCD	Charged Coupled Device. 6, 18
CIE	Commission Internationale de l'Eclairage. 17
DASL	Distance Along the Soil Line. 25, 26
DoLP	Degree of Linear Polarization. 7, 8, 63, 66–68, 117, 120, 147
DoP	Degree of Polarization. 9–11, 35, 42, 63, 64, 67–69
DTSL	Normal Distance To the Soil Line. 25, 26
ExG	Excess Green Vegetation Index. 45
ExR	Excess Red Vegetation Index. 45, 91
FLANN	Fast Approximate Nearest Neighbor Search Library. 86
FoV	Field of View. vi, 18, 19, 37, 38, 86, 91, 140, 145, 146
GUI	Graphical User Interface. 14, 22, 23, 99, 100
HSB	Hue Saturation Brightness. 17, 38
HSI	Hue Saturation Intensity. 17
HSL	Hue Saturation Lightness. 17
HSV	Hue Saturation Value. 17, 38, 39, 95
IPVI	Infrared Percentage Vegetation Index. 44
KKT	Karush-Kuhn Tucker conditions. 75
LIDAR	Light Detection and Ranging. 1, 3–5, 42
LoG	Laplacian of Gaussian. 51, 88, 90
LWIR	Long Wave Infrared. vii, 5, 7, 19, 20, 24, 27, 30, 39, 58, 59, 87, 141

MExG	Modified Excess Green Vegetation Index. 45, 91
MSRC	Microsoft Research Cambridge. 106
MWIR	Mid Wave Infrared. 24, 27
NDI	Normalized Difference Index. 45
NDVI	Normalized Difference Vegetation Index. v, 3, 18, 21, 42, 44, 45, 91, 92, 94, 106, 120, 144, 146
NDWI	Normalized Difference Water Index. v, 42, 43, 93, 94, 106, 120, 144, 146
NIR	Near Infrared. 3, 5–7, 18, 20, 21, 24, 26, 39, 42–45, 63, 86, 87, 91–93, 99, 103, 107, 120, 141, 142, 144
ORB	Oriented Binary Robust Independent Elementary Features. 86
PASCAL VOC	Pattern Anaysis, Statistical Modeling and Computational Learning Visual Object Classes. 15, 52, 54–56, 99
PDF	Portable Document Format. 23
POLAR	Polarimetric. 5, 7, 20, 21, 141, 142
PWM	Pulse Width Modulation. 86
RBF	Radial Basis Function. 22, 78, 85, 91, 96, 97
RGB	Red Green and Blue light. 6, 7, 15, 17, 18, 25, 26, 58, 99
ROC	Receiver Operating Characteristic. 80–82
RVI	Ratio Vegetation Index. 45, 146
SURF	Speeded-Up Robust Features. 86
SVD	Singular Value Decomposition. 51, 63, 87
SVM	Support Vector Machine. v, 21, 22, 71, 75–77, 96, 102, 104, 134, 137
SWIR	Short Wave Infrared. 5–7, 20, 24, 26, 27, 39, 42–44, 86, 87, 93, 94, 103, 107, 112, 139, 141, 142, 144
UGV	Unmanned Ground Vehicle. 1, 3, 20, 59, 137, 141
VIS	Visual Light. 3, 5, 6, 18, 20, 27, 38, 39, 42, 44, 45, 57, 59, 63, 86, 87, 95, 99, 139, 141, 142, 144

Chapter 1

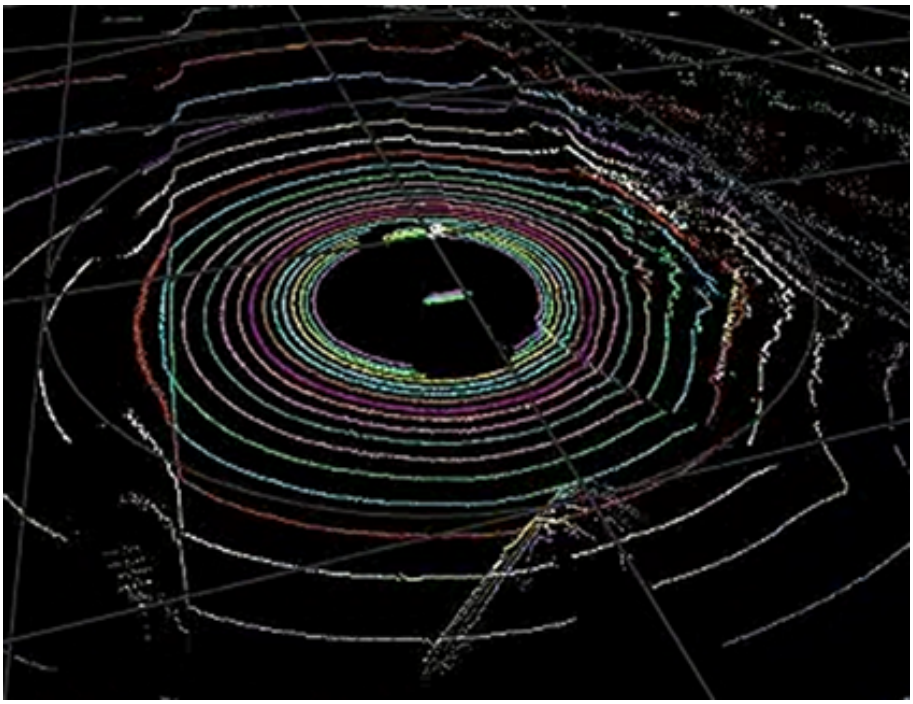
Introduction

1.1 Motivation

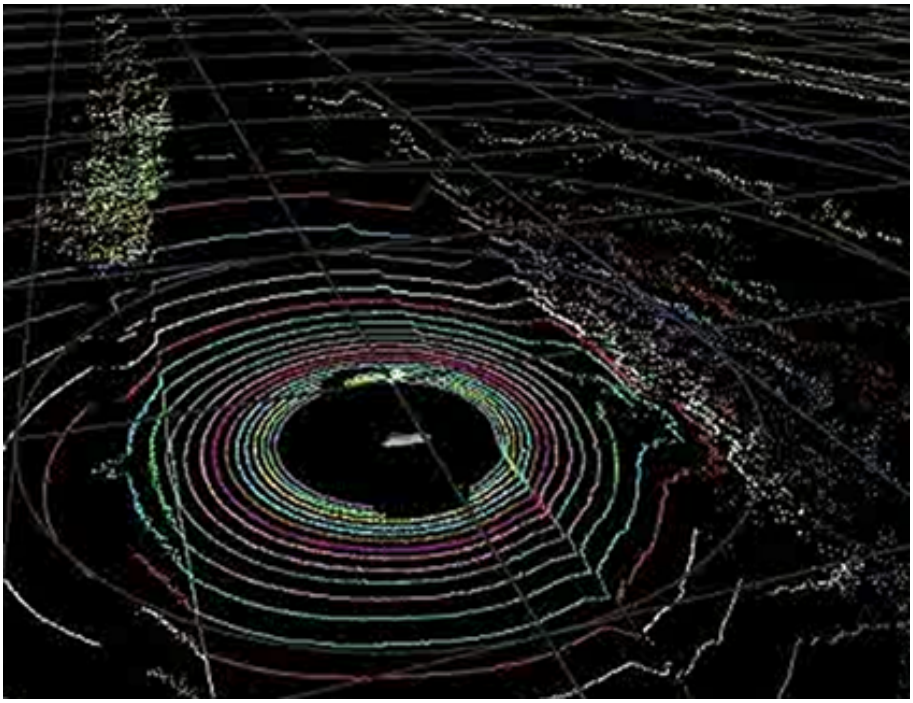
Previous work in the Mechatronics Lab at Virginia Tech has shown that while LIDAR is an important modality for perception of Unmanned Ground Vehicles (UGVs) certain objects can be difficult to detect when using Light Detection and Ranging (LIDAR) as the sole source of data. The Mechatronics Lab has explored multispectral imagery to compensate for these deficiencies. The current state of the art are two LIDAR units: the Velodyne HDL-32E and the HDL-64E shown in figure 1.1.1. The instruments measure distance using 32 or 64 planes, while spinning in 360°.

Figure 1.1.1: LIDAR systems from Velodyne Lidar [8]. Left, HDL-32E with 32 planes. Right, HDL-64E with 64 planes. V. A. Inc., "Velodyne Lidar," 2014. [Online]. Available: http://velodynelidar.com/. Used under fair use, 2015.

LIDAR data is very sparse and returns a small amount of measurements for a unit of area. This problem is exacerbated over long distances during which the rays diverge from the instrument and when the targets are thin objects. Tall grasses and guy wires are prime examples. Figure 1.1.2 is an example of data acquired from an HDL-32E where some objects can be difficult to identify due to this sparsity, especially without any context. One can identify that there is another vehicle below the sensor in 1.1.2a more easily knowing that the data was captured from atop a vehicle on a municipal road.



(a)



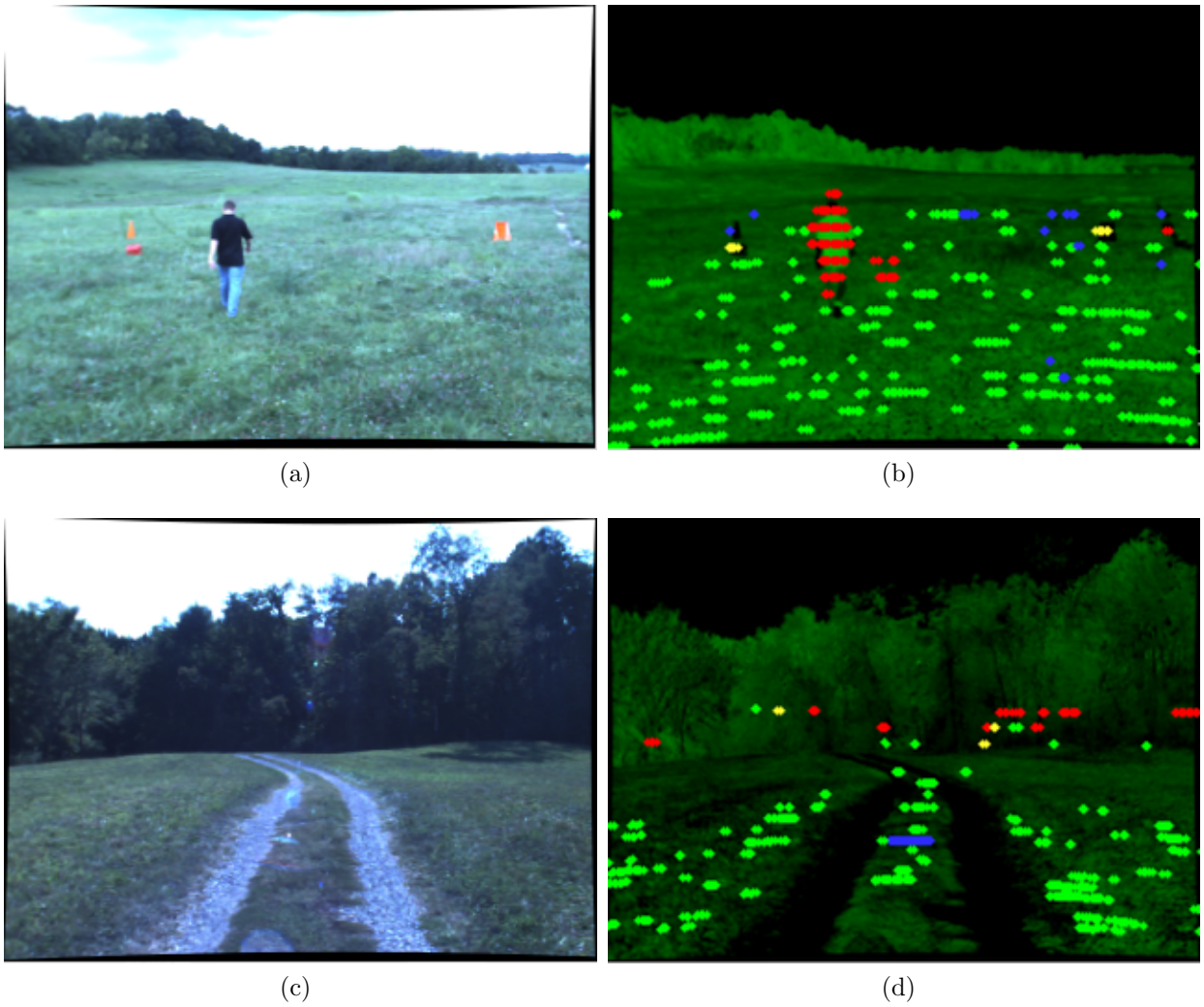
(b)

Figure 1.1.2: Example output of a 32 plane Velodyne HDL-32E, mounted on top of a consumer vehicle while driving on a municipal road. In 1.1.2a one can see another vehicle below the sensor. In 1.1.2b one can see the median and a tree to the left of the road.

However, what are the objects to the right of the page? Further down the road in 1.1.2b it is also difficult to classify the area beyond the road boundary. This makes sense intuitively when one considers that LIDAR returns are more similar to the sense of touch rather than sight. Flimsy objects can be confused with impassable objects if the appearance is similar in size, shape, and porosity. Again, fewer returns can exacerbate this effect.

Previous work in the Mechatronics Lab has found that sparse clumps of grasses can be a significant source of problems for LIDAR systems in rural environments. A large grassy field with vegetation ranging from 6-12 inches tall may contain clumps of grasses larger than the majority of the vegetation nearby. These obstacles have a consistent color and texture that humans can easily recognize. What if the UGV could use this information? The computer can also sense bands of light outside the visual spectrum so why be limited to just visual light? These motivations prompted the Mechatronics Lab to integrate imaging systems into an existing UGV, in order to supplement the existing LIDAR. The solution used two bands of light: Visual Light (VIS) and Near Infrared (NIR) which can be combined to produce a measurement sensitive to chlorophyll like responses. This combination is called the Normalized Difference Vegetation Index (NDVI)[3], see 2.1.3 for more information. The results of this previous work are shown in figure 1.1.3. Figures 1.1.3a and 1.1.3c show an original VIS image of problematic grassy areas. The corresponding images 1.1.3b and 1.1.3d show the application of a false color vegetation detection on the visual images. A measure of confidence is mapped to the green plane from 0: no vegetation to 255: completely vegetation. LIDAR points were mapped onto the frame of the camera and painted onto the image. The painted dots represent changes of the old classification system using only LIDAR to the new one using LIDAR and imagery. Green dots represent passable obstacles in both systems and red if both classify it as impassible. Blue represents a new passable object that was previously impassible and yellow dots are new impassible objects that were previously passable.

Figure 1.1.3a and 1.1.3b show an example where several clumps of grass are shown as passable in the new system (blue), and some traffic cones that are classified as new impassible objects (yellow). Likewise figure 1.1.3c and 1.1.3d show clumps of grass between the tire ruts that are new passable objects. These are examples were the new vegetation detection information was used to reduce the false positive impassible objects within grassy environments.



(a) (b)

(c) (d)

Figure 1.1.3: Previous work on LIDAR & camera fusion. Left: VIS output. Right: corresponding false color vegetation detection with object markers showing improvement from LIDAR as the sole sensor to fusion of LIDAR & vegetation detection. Green marker: passable objects in both systems. Red marker: impassable in both systems. Blue marker: previously impassable now passable with imagery. Yellow: previously passable now impassable with imagery.

Replacement of the LIDAR with camera systems is a long term goal for three reasons detailed here: 1) Camera systems are usually much cheaper than LIDAR, especially given the ubiquity of consumer electronics 2) Imagery is very dense compared to the inherently sparse LIDAR measurements and higher data density enables longer distances for object detection 3) LIDAR systems are active sensors that emit energy to perform the measurement, which enables other systems in the field to detect the bright signature of emitted light. In

particular many LIDAR systems are visible to normal consumer level cameras because NIR light is commonly used for the sensor of which consumer level cameras are sensitive to. The Velodyne HDL-64E uses $905nm$[15] for example.

The goal of previous the work was to supplement the LIDAR system to facilitate this long term goal. The implementation transformed the pixel information into the frame of the LIDAR, and then used the extra information as an additional features. This is possible because the LIDAR produces the distance information required to perform this registration. Since the original system was a classifier trained on the LIDAR data it was relatively straightforward to included the imagery once the registration was complete. The pixel data was appended to existing features and the classifier re-trained. The goal of the imaging subsystem then was to produce a measure of vegetation on the pixel level and pass that information downstream to the LIDAR classifier. This classification on the pixel level is termed the image labeling problem[16][17] or the semantic image labeling problem[18].

1.2 Image Labeling

Image labeling is to assign a class to each pixel from a finite set. The results can be expressed as an image mask where each color represents a class or label, as shown in figure 1.3.7. Image labeling is the combination of segmentation and classification. It is a segmentation problem as each pixel is grouped and a classification problem because each group is assigned a label.

Image classification can be performed at different levels of abstraction: image, sub-image, and pixel level. Classification on the image level means to assign a label on a per image basis, e.g. a target is present in an image for positive hit. Bounding-box or sub-image level describes the target class by a center point and rectangle that contains the object. Pixel level is where a class is assigned at every pixel. Further discussion regarding computer vision and machine learning terminology can be found in 1.3.4.

Image labeling is a standard task in computer vision. Systems that perform image labeling can be used to automatically annotate imagery for retrieval given a keyword[19][20] and detect interesting targets[21][14] for example.

1.3 Background

1.3.1 The Hardware Platform

Previous work in the Mechatronics Lab created the multispectral rig shown in figure 1.3.1. The bands sensed are VIS (4μ - $7\mu m$), NIR ($7\mu m$ - $11\mu m$), Short Wave Infrared (SWIR) ($0.9\mu m$ - $1.7\mu m$), and LWIR ($8\mu m$ - $14\mu m$), and Polarimetric (POLAR) within the visi-

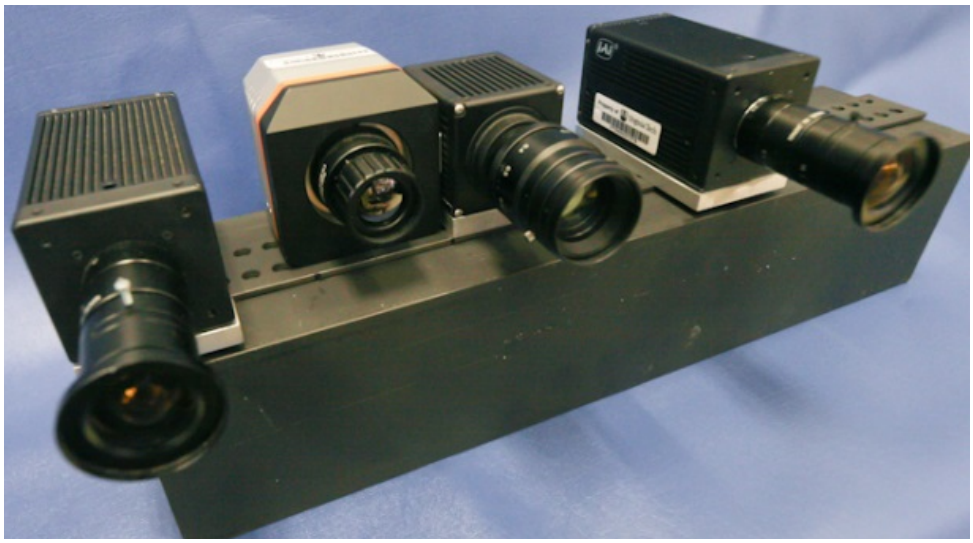


Figure 1.3.1: The multispectral rig used by the Mechatronics Lab at Virginia Tech. From left to right: JAI AD-080GE(VIS/NIR 0.4μm - 0.65μm, 0.76μm - 1.0μm)[22], Gobi-384(LWIR: 8μm - 14μm)[10], SU320HX-1.7RT(SWIR: 0.9μm - 1.7μm)[23], a second JAI AD-080GE, Polarimetric VIS/NIR camera (not shown) created by the Mechatronics Lab[6] using 4 Point Grey FMVU-03MTM/C(VIS/NIR 0.42μm - 1.0μm)[24].

ble/near infrared band (4.2μm - 10μm)

The JAI cameras use a beam splitter, a dichroic prism, to separate the signal into two channels: VIS & NIR[22]. The beam splitter reflects half of the incoming light and passes the other half. Two Charged Coupled Devices (CCDs) then are able to sense the same optical plane: one with a Bayer filter for Red Green and Blue light (RGB) (a.k.a VIS) and the other with a bandpass NIR filter. The images are optically aligned as a result meaning that the two frames can be overlain on top of one another and each pixel represents the same point in space. Stereo is possible using the two JAI cameras but is not considered in this study.

This thesis measured polarimetric information in the visible and near infrared bands using a camera created by Mark Umansky of the Mechatronics Lab[6]. The polarimetric camera uses four USB cameras, three beam splitters, and four polarizing filters to sense the images necessary to calculate features from polarization, discussed briefly below and in depth at 3.2.3.

An example of the data available from this rig is shown in 1.3.2. The appearance of objects within the bands shown is a continuing discussion throughout this thesis and will be addressed in depth. For introductory purposes however there are few key phenomena that should be noted. Vegetation in the NIR band is very bright and more texture is present as one can see from the vegetation and clouds in the background. The SWIR band shows a dark sky and bright vegetation. The glass in the the foreground vehicle in SWIR is much darker compared to the VIS and NIR. Data from this work has generally shown that vehicles

and buildings have especially dark windows although normal glass does pass SWIR light. The LWIR band does not pass through glass however and so is opaque. The LWIR band is temperature dependent, so the sky is relatively dark and clouds are not as stark. Vegetation is normally cooler than the ground where the asphalt and concrete in this case are much hotter in the midday sun. Note that the white roadway lines are visible in the LWIR. Lastly the polarimetric data is previewed in 1.3.2e and 1.3.2f. One of the polarimetric cameras and the Degree of Linear Polarization (DoLP) is shown because the DoLP is more informative than the 4 raw feeds. The POLAR cameras sense between $0.42\mu m$ - $1.0\mu m$[24] and so are similar to NIR. The DoLP is interesting as it accentuates reflections and man made materials. Again, this is discussed further in 3.2.3 Polarization Features and 2.1.2 Water and Sky Detection.

1.3.2 Multispectral, Hyperspectral, & Polarimetric Imaging

Concepts

Multispectral and hyperspectral imaging differ by the number and breadth of frequencies sensed. Multispectral data measures at least two frequency bands numbering on the order of 1 or 10 while hyperspectral is on the order of 100[25]. The *frequency band*, or *spectral range*, or simply *band*, is an interval of continuous wavelength[26]. For example, the visible band is considered a multispectral band: typically spanning from $400nm$ *to* $700nm$ with three sub bands for RGB. Multispectral measures the aggregate intensity within wide and separated bands, on the order of 100 nm, and hyperspectral measures intensities at very narrow bands across a wavelength range[26]. Airbone Visible/ Infared Imaging Spectrometer (AVIRIS) for example measures 224 channels with $10\mu m$ resolution and a total frequency range of $380nm - 2500nm$[26]. When multispectral or hyperspectral images of size $[m \times n]$ are arranged in a vector of size k, the resulting 3D matrix is termed a *data cube* or an *hyperspectral/multispectral cube* of size $[m \times n \times k]$. Pixels at the same location $[i < m, j < n]$ represent the same physical point in space sensed at the respective band k. Hyperspectral data is also referred to as "imaging spectroscopy" and "imaging spectrometer data"[27].

Polarimetric imaging is a measure of the proportions of polarized and unpolarized light. Light is a transverse wave constructed of an electric field $\vec{E}$ and a magnetic field $\vec{B}$ perpendicular relative to one another and the direction of travel: the direction of the $\vec{E}$ axis[28], shown in figure 1.3.3.

(a) VIS

(b) NIR

(c) SWIR

(d) LWIR

(e) Camera 0 source for polarimetric camera

(f) Degree of Linear Polarization (DoLP)

Figure 1.3.2: Example data available to this thesis, showing a parking lot. Raw imagery is shown with the exception of DoLP image computed using the polarimetric camera, which highlights reflective surfaces. See 3.2.3 for details.

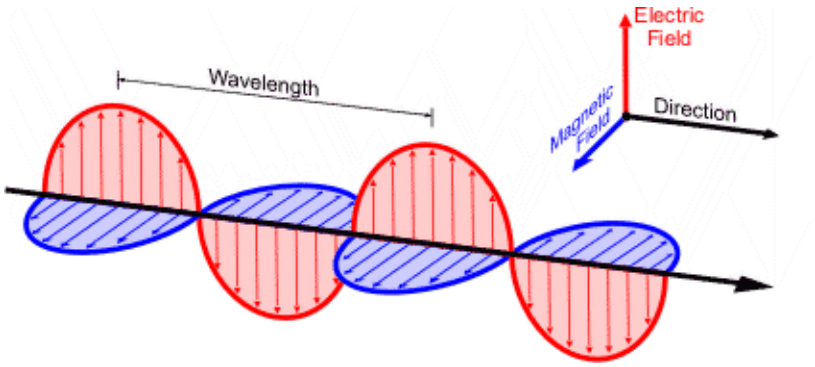


Figure 1.3.3: Components of an electromagnetic wave. A transverse wave: an electric field and magnetic field perpendicular to one another and the direction of travel[28]. N. W. Service, "National Weather Service Remote Sensing Introduction," 2014. [Online]. Available: http://www.srh.noaa.gov/jetstream/remote/remote_intro.htm. Used under fair use, 2015.

Light has three components pertinent to this study: *intensity*, *partial polarization*, and *phase*. The direction of travel is the direction of energy transfer, or intensity. The magnitude of $\vec{E}$ and $\vec{B}$ are combined to form the electric vector[29], where the polarization orientation is the angle of that vector. This orientation is also known as the *polarization phase*. This angle is measured with respect to an arbitrary axis but can be described relative to gravity. "The relative portion of linear polarization [is] *partial polarization*", also known as Degree of Polarization (DoP)[5]. This is shown in figure 1.3.4.

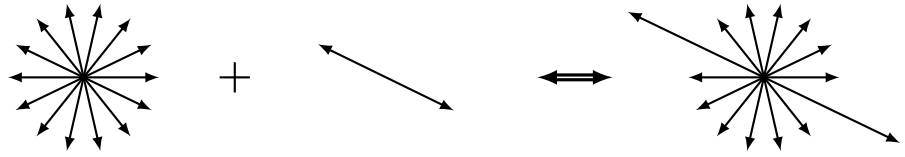


Figure 1.3.4: Partially linearly polarized of light is the superposition of unpolarized and linearly polarized light. Shown in the direction of travel of the waves. Adapted from [5]. L. B. Wolff, "Polarization vision: a new sensory approach to image understanding," Image and Vision Computing, vol. 15, no. 2, pp. 81-93, Feb. 1997. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0262885696011237.

Completely polarized light will have a partial polarization value of 1 as it is described in terms of a percentage. Completely unpolarized light is where the light has "sudden changes in angle and orientation with no correlation between [orthogonal] components [of the polarization angle]" which can change over time[6]. Another valid interpretation of DoP is that some unpolarized light has been superimposed on fully polarized light.

There is a fourth component of light that describes elliptically or circularly polarized light. In this case the *polarization phase*, or the angle of the electric vector, is continually changing with respect to time. When the amplitudes of the two fields $\vec{E}$ and $\vec{B}$ are equal and 90° out of phase the light is circularly polarized; it is elliptically polarized if the amplitudes are unequal and 90° out of phase, or if the relative phase is not 90°[29]. This component is not always measured in polarization cameras as circular polarization "does not occur in many places in nature"[6] and measuring DoP alone can be sufficient[5]. This is discussed further in 3.2.3.

Polarization by Reflection

Light becomes linearly polarized or partially polarized upon reflection. This is shown in figure 1.3.5 where incident light is reflected off the surface between the two materials. The plane of incidence is the plane created by the vector of the propagation direction and the normal vector of the reflecting surface (shown in figure 1.3.5 as the plane of this page). The angle of reflection is measured from the normal vector of the reflecting surface. Light becomes linearly polarized upon reflection when the angle θ_B is equal to the Brewster's Angle, and becomes partially polarized at all other angles[29]. The reflected polarized light has an electric field vector pointing in the direction perpendicular to the plane of incidence, and parallel to the surface of reflection. The refracted light is partially polarized. The value of Brewster's angle is dependent on the indices of refraction of the two materials.

This effect is important for material classification as man-made materials, water reflections, and the sky have singular responses. Shown in figure 1.3.6 is a polarization contrast image highlighting asphalt, the brick of a building, and a vehicle. People often observe this effect when using polarizing sunglasses to reduce such glare. The light becomes partially polarized upon reflection and the sunglasses can reduce the glare off the such surfaces. The polarization contrast shown in 1.3.6 is a simplified calculation similar to DoP discussed in 3.2.3 Polarization Features.

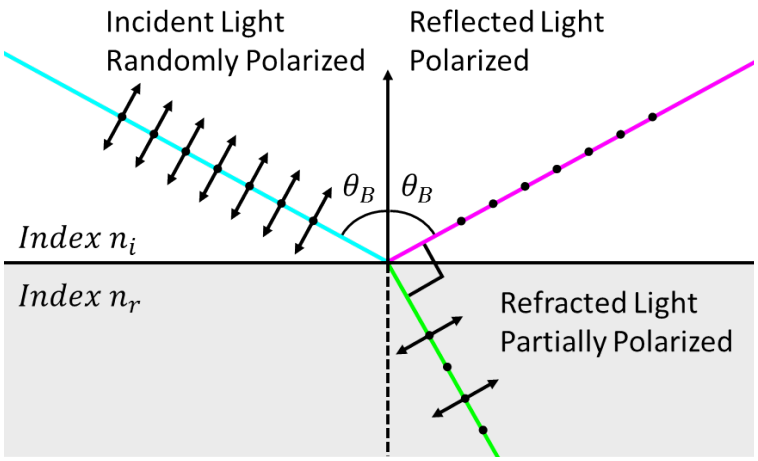


Figure 1.3.5: Polarization by reflection showing the angle of incidence. This angle becomes Brewster's Angle at the point of zero reflection of light parallel to the plane of incidence. The plane of incidence is the plane of this page. Adapted from [29] C. R. Nave, "Classification of Polarization," 2012. [Online]. Available: http://hyperphysics.phy-astr.gsu.edu/hbase/hframe.html. and [30] I. Wikimedia Foundation, "Brewster's Angle," 2015. [Online]. Available: http://en.wikipedia.org/wiki/Brewster%27s_angle.

Figure 1.3.6: Example data showing polarization by reflection. Primary targets are asphalt, a building, and a vehicle. Left, image from a horizontal polarized filter. Right, the polarization contrast between horizontal and vertical polarized images[6]. M. Umansky, A. L. Wicks, K. Meehan, and D. W. Hong, "A Prototype Polarimetric Camera for Unmanned Ground Vehicles; Thesis submitted to the faculty of the VPI&SU in partial fulfillment of the requirements for the degree of Master of Science in Mechanical Engineering," Virginia Polytechnic Institute and State University, Tech. Rep., 2013. Used with permission of Mark Umansky, 2015.

A possible source of confusion for these polarization measurements is that the DoP values can be similar in shadows[4], which would explain why the vegetation is bright in 1.3.2f.

1.3.3 Introduction to Machine Learning

Learning from Data

Machine Learning is the application of computer algorithms to automatically learn patterns from data. It is a very broad subject that is related to stochastic theory, data mining, artificial intelligence, and pattern recognition. It has a myriad of applications within for example the realm of science and commerce including drug discovery, surveillance, speech & text recognition, and most pertinently: computer vision, robotics, and autonomy[13][31]. The goal of machine learning is to apply these learned patterns to "predict future data or other outcomes of interest"[13].

For example, Netflix rents out movies to its customers and has a system that recommends new movies based on what the customer has watched and rated[32]. Netflix offered 1 Million dollars to whoever could improve their recommendation system by 10%[32]. The question is, what rating will a customer give a movie they have not yet seen? One method would be to interview the customers to define their tastes, watch all the new movies and define characteristics, then compute a similarity between the two[33]. This would however be time intensive and expensive. Machine learning instead starts with the rating and then tries to determine the model that explains the data[33]. One can think of this as: what movies have other people liked that have similar tastes to the new person in question?

There are three characteristics that should be present before one applies machine learning: "1) A pattern exists 2) We cannot pin it down mathematically 3) We have data on it"[33]. If no pattern exists then subsequent attempts to find one will fail. If one can write out an equation by hand, then that would be faster than using a learner. However, if no data exists then no machine learning can be done.

Supervised, Unsupervised, and Reinforcement Learning

Machine learning is divided into two main styles: supervised and unsupervised learning. The supervised method uses data that has been previously labeled as belonging to a particular category to predict the category that unlabeled data belongs to. It aims to produce the most accurate predictions[31] and acts on labeled data to map the "inputs x to outputs y given a labeled set of input-output pairs"[13]. Future data points are not in the original data set but are "assumed to be generated by the same unknown process"[31]. Unsupervised learning does not have labeled data to train on and instead focuses on discovering the patterings and describing the data. Clustering for example groups similar data points together so that the model has a more simple representation. The lack of labeled data prevents an error metric

from being used, like in supervised learning where one can compare the prediction results to what happened in the data set. The unsupervised problem is much less defined but does not require the often expensive process of labeling data.

Reinforcement learning is a third style but is not as common. The learner aims to maximize the probability of a long term goal but actions must be taken at successive steps but a reward or punishment signal is only occasional[13]. For example, playing a game requires actions at each turn but one does not know the result until the end. Reinforcement learning is often is a more advanced topic in machine learning, sometimes being out of scope of large pedagogical works on the subject[31][13].

Given the measured variables x and the predicted variable y, a machine learning problem is generally written in the following manner: data D is a set of N paired inputs and outputs $D = \{x_i, y_i\}_{i=1}^{N}$[13]. The feature vector x_i can be m dimensional and typically is large. In this thesis for example a feature vector describing the pixels can be the intensity values, such as red, green, blue. When the output value y is categorical, or finite, the problem is termed *classification* where $y \in \{1, \ldots, C\}$. If the set of classes is continuous then the problem is termed *regression*. In unsupervised learning the data no longer has a class and is simply $D = \{x_i\}_{i=1}^{N}$.

Discriminative vs. Generative

There are two schools of thought that exist when creating a learning algorithm: discriminative and generative. The distinction is on whether the algorithm models the data D or the boundary separating the classes. A generative approach models the joint probability of the class and data $p(y, x)$, and then afterwards predicts by deriving the conditional probability $p(y|x)$. This effectively asks the question: knowing how the data is distributed, what is the probability that this data point belongs the class y? The descriptive approach instead models the joint probability $p(y|x)$ without reference to how the data is distributed. There are many pros and cons to each approach, however in general descriptive algorithms require less data to train and less prior knowledge of the problem but are more complicated in their structure and method of training.

Application of Machine Learning Concepts

These styles of learning are highlighted because one must carefully choose the type and family of algorithms based on the nature of the problem in order to maximize success. Certain learning styles and algorithms will be more effective than others, while some may not be applicable at all. Labeled data for instance is often a limiting factor. Hand labeling imagery is time intensive, expensive, and large amounts are needed to train algorithms properly[34]. "In many real-world classification scenarios, it can be much easier to collect input data than to observe associated labels, which could require relatively expensive human

action"[35]. In these cases, an active learner, or active search, is applicable[35][34]. Active search is a derivative of supervised machine learning where "... the learner interactively chooses which data points to label...contrasted to the passive model of supervised machine learning where all the labels [are known beforehand]"[36]. Drug discovery is an example of a good candidate for active search because it can be time consuming to model and test the chemical interaction, so it is beneficial to test the best candidates. Likewise, labeling may not be possible because one does not know what classes exist, or they are simply trying to gain insight to the problem. Unsupervised classification then would be the best option.

Supervised machine learning was chosen for this problem due to the ease of implementation: there is a significant amount of research in the area and many programming libraries are available on the subject, such as [37][38][39]. A Graphical User Interface (GUI) was created in order to label the data, which when paired with features creates labeled samples. The system was on a pixel level so that the number of samples was actually quite high: 5.024×10^6 labeled pixels total. This however does not necessarily approximate the number of samples well as the pixels are highly correlated with one another. As a result the data was split on an image basis rather than pixel basis, of which there were 133 frames in the entire set. The supervised machine learning methods chosen impacted development in that hyperparameters were required, see 5.1 Hyperparameters. Unsupervised machine learning methods were added in order to incorporate texture into the classifier, although this decision was not made due to machine learning factors, rather, the textural information was driving force. See section 3.2.1 Texture.

1.3.4 Concepts in Computer Vision and Machine Learning

Recognition, Detection, Classification, Segmentation

Within the fields of Computer Vision, Pattern Recognition, and Machine learning there exists terminology to describe the type of input and outputs to and from algorithms. Those pertinent terms are: Image Fusion, Classification, Detection, Recognition, and Segmentation.

Image Fusion, or *Data Fusion*, is the combination of multiple data sources to produce new or improved information. Multiple viewpoints or wavelengths of imagery are common examples of this: humans use stereo information from two viewpoints to sense depth and wavelength from 3 bands to sense color. Image Fusion often occurs through images that differ in time (video), space (viewpoint), and color (wavelength). Formally, "Image fusion is the combination of two or more different images to form a new image by using a certain algorithm"[40]. There are three levels of image fusion that describe the data input: 1) Pixel, 2) Feature, 3) Decision level[40][41]. Pixel level fusion is performed directly on the intensity information. Feature level is fusion "requires the extraction of objects recognized [in the data]"[40]. It is performed after feature extraction. Examples of features include descriptions of edginess, such as a Laplacian of Gaussian filter which emphasizes regions in the image that change

brightness quickly between pixels. Decision level, or Symbol level, signifies that a decision has be performed. Does a certain object class appear in an image, for example? Decision level fusion is the highest level of abstraction that the data can be represented in, and requires a low level of correspondence between the sources of information[42] e.g. two algorithms could have detected the presence of a class in some data and one can be more confident in the classification using two independent sources. *Classification*, *Recognition*, *Detection*, and *Segmentation* similarly can be categorized based on a level of abstraction.

Classification "is a process that assigns a label to an object" and a classifier outputs said class given the object representation[43]. While definitions vary slightly, it refers to predicting if an object of a class does or does not exist in an image. Image Classification typically deals with output on the image level, i.e. the entire image is classified, while classification can be performed on any pairing between categorical or nominal outputs and the corresponding features.

Detection aims to identify where an object exists in an image, drawing a bounding box around the target for example. It is also known as object localization[13]. This style of image processing outputs on the sub-image level, i.e. a cropped portion of an image.

Recognition attempts to differentiate between object instances to identify the user or owner of said object. Face detection for example would output the name of the person shown.

Image Segmentation divides an image into a set of regions. The regions are grouped by a common feature and/or semantic meaning. A combination of edges, lines, and other shapes may be the desired characteristic. It may have a general scope such as foreground/background, or a more complicated one such as specific animals and objects. Formally, image segmentation has two goals: decomposing an image into parts and representing the regions at a higher level of abstraction[43]. Images can have a large number of pixels so distilling the image down to a smaller number of super pixels can reduce the processing requirements downstream while still maintaining the essential information[44]. A form of segmentation is performed by humans to produce pixel level training data for supervised machine learning. The image may be termed as *'painted'* if a human draws colors over an image representing individual classes. This is shown in an example from the Pattern Anaysis, Statistical Modeling and Computational Learning Visual Object Classes (PASCAL VOC) 2011 challenge in figure 1.3.7. Another keyword is *annotated*, which means that the image has been processed to have some level of classification, e.g. 1.3.7a has been annotated by a human at the pixel level to produce 1.3.7b.

Color Space

Images are often encoded in a three dimensional RGB color space during image processing and display operations, with 8-bit precision per channel. There are other color spaces and encoding formats that have been optimized for specific devices, human consumption, or

(a) (b)

Figure 1.3.7: Example image segmentation. The segmentation is hand labeled, or painted, showing the object classes of human, motorcycle, and void for the object boundaries[45]. M. Everingham, L. Van GooL, C. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results." [Online]. Available: http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. Used under fair use, 2015.

computational efficiency. It is can desirable to use a color space other than RGB as differences in colors and brightness can be distinguished more easily. The purpose of this passage is to introduce the terminology necessary for understanding the pertinent color spaces used in this study. Mathematical transformations between color spaces are withheld here. The terminology begins with: *Luminosity*, *Photometric* vs. *Radiometric*, *brightness*, *hue*, and *saturation*.

Luminosity, or *Luminous Flux*, is the perceived light intensity of a visible light source[29][46] and has the SI unit of lumen ($lm = cd \times sr$)[29]. *Photometry* and *photometric* measurements "seek to describe quantitatively the perceptual brightness of visible [light to humans]"[46]. *Radiometric* measurements on the other hand seek to describe the absolute or physical measurement. The distinction is important because humans perceive the brightness of a source of light as a function of the wavelength. The *photopic luminosity function* describes the relative brightness of light with respect to wavelength, during high levels of light (daytime). This sensitivity with respect to wavelength is termed *luminous efficacy*[29]. Humans perceive light differently in low levels (nighttime) so there is another function called the *Photopic luminosity function* that describes this effect in low light levels. Humans are most sensitive to green light ($555nm$) during high levels of light[29].

"Often the luminance of a color is not of interest in a color match"[46]. *Luminance* has the unit of $cd/m^2 = lm/sr/m^2$ and is a measure of the perceived "source intensity per unit surface area"[47]. *Intensity*, or *radiance* is the analogous radiometric term. This is important to this work because the perceived brightness of a color is often a confounding variable when

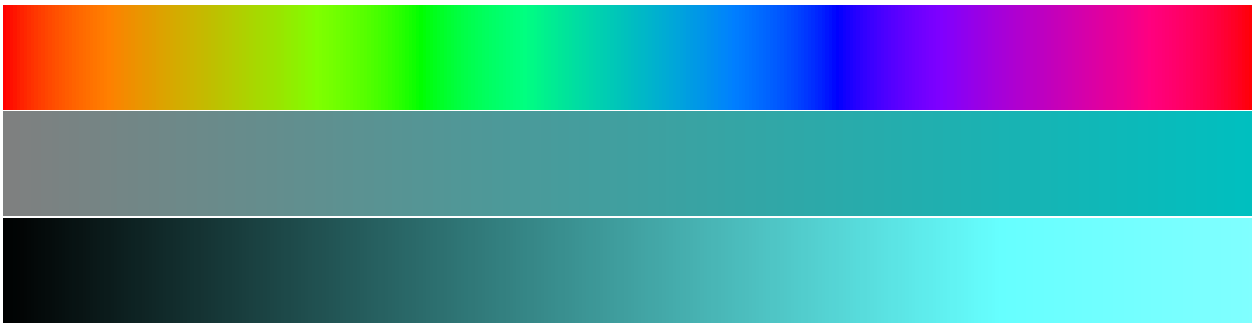


Figure 1.3.8: Example of the HSI color space. The Hue dimension is shown on top, Saturation on the middle and Intensity on the bottom.
The hue bar varies $H = [0°, 359°]$ left to right $S = 1, I = .5$.
The saturation bar varies $S = [0, 1]$ left to right, $H = 180°$ (cyan), $I = 0.5$.
The intensity bar varies $I = [0, 1]$ left to right, $H = 180°$ (cyan), $S = 0.5$.

identifying and object by said color. Shadows for instance can be a source of confusion for a classifier. When one describes a color irrespective of intensity (or luminance) only two terms are needed to describe a color match[46]. These are named *hue* and *saturation* in the color spaces of Hue Saturation Intensity (HSI), Hue Saturation Value (HSV), Hue Saturation Brightness (HSB). *Brightness* is considered the third component of light that describes the luminance, but does not have an SI unit[29].

HSI is a color space common to computer vision applications because it can normalize for lighting and separate the color components that describe the surface from the source that is lighting it. An example of HSI is shown in figure 1.3.8. HSI is similar to HSV and Hue Saturation Lightness (HSL). The latter two vary slightly in the third variable that describes brightness. HSB and HSV refer to the same model whereas the term HSB is used more often in Adobe Photoshop[48].

Hue **H** describes the wavelength or combinations of wavelengths that create the color. Gray light has a zero hue value, and white and black can be produced with any hue value. Magenta for example is created by combining the spectral colors red and blue[29]. Hue is measured between 0 and 2π, with red at $0°$.

Saturation **S** describes the purity of a color from 0 to 1. A completely unsaturated hue (0) is gray, and a completely saturated hue (1) is pure.

Intensity **I** spans from black (0) to white (1). It is similar to the RGB cube diagonal from [0,0,0] to [255,255,255][43].

Another color space is 1976 $CIE\ L^*a^*b^*$ Space ($CIE\ L^*a^*b^*$). Commission Internationale de l'Eclairage (CIE) "is an international body concerned with standards for light and color" and has established said $CIE\ L^*a^*b^*$ space[46]. It was created with the goal that the perceptual differences between colors were proportional to the Euclidean distances in the L^*, a^*, b^*

space[49]. *CIE* $L^*a^*b^*$ is used when a "perceptually uniform"[14] space is required. L^* spans $[0, 100]$ with 0 being black and 100 as white. An example of this coordinate system is shown in figure 1.3.9, where the axis a^*, b^* are given colors at $L^* = 50$. The positive and negative scales represent opposite colors as there cannot be both red or green (a), or both blue and yellow (b) within the same color[50].

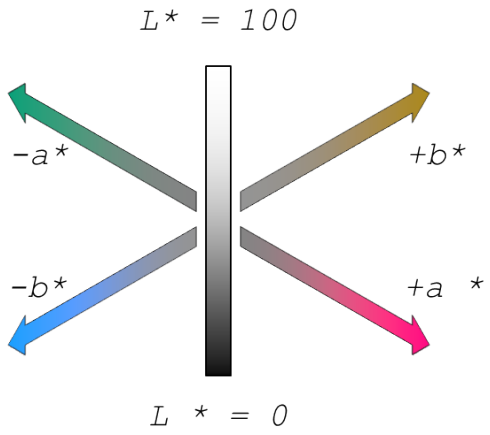


Figure 1.3.9: Example of the *CIE* $L^*a^*b^*$ color space. Axes a^*, b^* are shown with colors at $L^* = 50$. Adapted from [50] "Technical Guides Color Models CIELAB," p. 2, 2000. [Online]. Available: http://dba.med.sc.edu/price/irf/Adobe_tg/models/cielab.html.

Image Registration

Image Registration is a cornerstone of this study as it allows the fusion of images from different viewpoints. Image registration is the process of spatially aligning images, e.g. of size $[m, n]$, so that the pixels at the same image location $[i < m, j < n]$ represent the same physical point in space. This was discussed briefly in 1.3.2. Figure 1.3.10c shows a toy example of image registration. Two images where captured using the JAI VIS/NIR sensors of the same target but different angles. Figure 1.3.10a shows two VIS images that have been registered to one another. Note the black border that shows the limits of Field of View (FoV). Figure 1.3.10b shows a NIR image that has been registered to the VIS on the upper left, shown in grayscale now. The pixel intensities from the VIS band can now be fused with the values of the NIR band as the images have been spatially aligned. Figure 1.3.10c shows the thresholded results of the vegetation detection algorithm NDVI painted green. The operation can be performed on the overlap. Please also note that the JAI cameras produce registered RGB and NIR because of the use of two CCDs and a beam splitter as explained in 2.3.2, and image registration is not necessary. The toy example was for illustration. This type of image registration is termed *mosaicking* as several images are arranged into one large image, as if one was moving tiles. *Panoramas* are mosaics created by taking images at one point but twisting the camera over a large angular span.

(a) VIS/VIS mosaic (b) VIS/NIR mosaic (c) VIS/NIR mosaic with NDVI

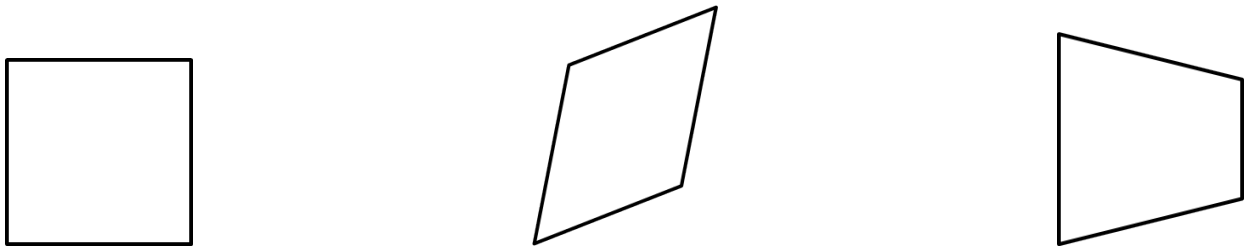
Figure 1.3.10: Toy example of image registration between pairs of VIS/VIS, VIS/NIR. Multispectral fusion is enabled when the images from different bands are registered. A binary NDVI output is painted in green over the registered portion.

Not all cameras can be registered through the use of beam splitters however. LWIR for example cannot pass through the glass used for normal optics, and beam splitters may add constraints on the FoV[6] and/or be cost prohibitive. When separate cameras must be used images must be registered. Several terms are defined here in order to discuss this using the terminology used by Goshtasby[51]: *Reference Image*, *Sensed Image*, *Transformation Function*, *Affine Transform*, *Projective Transform*.

One image is termed the *reference image* as it is not changed, and is also known as the *source* image. The other image is termed the *sensed image*, or target image, which is warped so that it aligns with the reference image. The sensed image is warped using a reference image.

Transformation functions vary by the number of parameters needed to define the coordinate transform. Three most basic transformations listed in increasing complexity are: translation, rotation, and scaling. Translation is a displacement along the image axes, rotation twists the entire image about one point, and scaling is the same as zooming in and out. The transformations operate by matching pixel locations from the sensed image to the reference. The number of points required is dependent on the degrees of freedom defined by the transformation function.

The *affine* transformation combines the capabilities of translation, ration, and zoom, and adds shearing. Parallel lines are preserved in affine transformations[52]. These transforms are applicable when the scene is very far away from the camera[51], meaning that the disparity produced by the displacement of the viewpoints is small. Such disparity is inversely proportional to distance and proportional to focal length. The affine transform has 6 degrees of freedom and requires 3 non-collinear points matched between the sensed and the reference image to perform[51]. The *projective* transformation, or homography, is the next level of complexity with 8 degrees of freedom and requires 4 matched points to perform, three that are not collinear. Projective transformations require that straight lines are preserved be-



(a) Reference (b) Affine (c) Projective

Figure 1.3.11: The effects of an affine and projective transform. Adapted from [52]. R. Szeliski and U. Szeliski, Richard (Microsoft Research, "Image Alignment and Stitching: A Tutorial," Foundations and Trends in Computer Graphics and Vision, vol. 2, no. 1, pp. 1-104, 2006. [Online]. Available: http://www.nowpublishers.com/product.aspx?product=CGV&doi=0600000009.

tween the reference and sensed images[51]. The warping capabilities of affine and projective warping are summarized in figure 1.3.11.

The focus of this thesis is on the multispectral fusion for applications involving UGVs. Image registration is a required stepping stone to that goal but is not explored in depth. There are many problems within image registration that one could compose a thesis to address, such as LWIR/VIS registration[53]. This is because traditional registration techniques compare image intensities directly when matching the sensed image pixels to the reference image. These techniques are not as successful here as intensities between bands vary widely for the same materials, see figure 1.3.2. Indeed, that is the cornerstone of this thesis: multispectral imagery provides unique information. This however complicates the registration process. This thesis employs homography algorithms from the OpenCV C++ library[39] to perform the necessary image registration, with minor changes to hyperparameters to accommodate the nature of the imagery. Unfortunately LWIR was not included in this work as such registration requires a significantly different approach. SWIR fortunately is similar enough to register to the NIR band of which was already registered to the VIS band via the beam splitter. The POLAR images were measured in the combined VIS/NIR and so the original techniques were also applicable.

1.4 Thesis Overview

1.4.1 Problem Statement

This study seeks to solve the image labeling problem applied to the imagery obtained from the multispectral camera rig of the Virginia Tech Mechatronics Lab . The goal is to produce a complete system starting from capturing the data all the way to the analysis of the machine

learning algorithms applied. There are several key capabilities that must be addressed in order to produce such a system: 1) data collection 2) image registration 3) image annotation 4) feature production 5) learning algorithm implementation 6) performance metric analysis 7) learner training and hyperparameter selection. While a complete optimization of the system is outside the scope of this study, some optimization is performed during hyperparameter selection.

A secondary objective was to include multispectral and textural information as features. Multispectral information meaning more than simply including the intensities of the bands directly, rather the bands are combined in a way that new information is developed, e.g. NDVI. The learner can infer these combinations to some degree, but certain mathematical relationships may not be encoded implicitly and so are included in the feature list. The textural information was considered given the success of its use for material classification [54]. Material classification is an excellent source in literature as the classes considered are more similar to materials than objects, primarily due to the pixel level classification.

1.4.2 Thesis Summary

Chapter 2 Literature Review discusses strategies in perception with an emphasis on imagery in ground vehicle settings. Strategies are shown using multispectral and polarimetric data for targets likely to be seen by unmanned ground vehicles: mud, water & water reflections, sky, and vegetation. Preliminary data is shown either for raw imagery or for the features described in the literature. This can provide insight on the type of data or information being sought. General machine learning practices with regard to the learning curve and similar applications in literature are also discussed briefly.

Chapter 3 Theory discusses the mathematics for image registration, multispectral features, textural features, the learning algorithms used, and necessary steps for supervised classification. While the image registration step is a vital component of this study it is not optimized here and as such a brief explanation with high level pseudo code is presented. Similarly, full derivations are given only for the Support Vector Machine (SVM) as it is the core component of this implementation. Mathematical theory for textons and POLAR features are included as the application of filters and the phenomena behind POLAR data is fundamental. The other features used rely heavily on complex interactions between the light and the targets such as NDVI, or, merely were phenomena that were found to be informative such as Saturation/Brightness. NDVI for example relies on "high internal leaf scattering" of NIR and "chorophyll absorption" of red and blue[55], and explaining precisely why this interaction occurs would be out of scope. Indeed if one could explain in precise mathematics how all these interactions occur there would be no need to learn the pattern through machine learning. Information on how to apply supervised machine learning algorithms is also included here as there is not necessarily one solution, and it is fundamental to the application.

Chapter 4 Implementation discusses methodology required to implement this study espe-

cially if it was not addressed in Literature Review or Theory. The chapter addresses: image registration, feature production, classifier training, and SVM training, and the creation of the labeled dataset. The chapter is concluded with a list of the target classes and features used. Notes on each item is as follows. A high level pseudo code is provided for the image registration as this topic was necessary but not optimized for this work. The filter implementation shows the functions used to create the filters, the actual filter kernel images used, and the application map. The map shows how the said filter kernels and the source imagery were paired in literature. The texton implementation is explained briefly in order to detail how it was implemented for this work. Additional features then follow, for when specific implementation details are required to reproduce this feature set if such details were not included during the literature review and theory stage. Finer details for the specific SVM implementation are included which are the SVM kernel selection, the grid search concept used for SVM hyperparameter selection, and the expansion of the C-Support Vector Machine (C-SVM) classifier to address the class imbalance problem. Next the creation of the data set is addressed. Most importantly the reasoning behind the labeling scheme chosen is explained, which was to primarily reduce label noise and reduce the effort required to produce the labels. A GUI and C++ framework was created in order to label the imagery and the capabilities of this is briefly listed. The details are brief as this stage was a tool and not the focus of this work. However, it represented an invaluable learning experience and was crucial to the software produced later in the process. The strategy used to partition the datasets is addressed here, as it is very important to produce repeatable experiments as well as prevent the classifier from testing on the same training data. To this end images were partitioned into test/train rather than on a pixel basis. The aforementioned feature list concludes the chapter and details the name of each feature and the representations of the data.

Chapter 5 Experiments discusses the experiments conducted, the methods for the analysis of said experiments, and the hyperparameters required to perform them. The parameters C, γ are required for the C-SVM and Radial Basis Function (RBF) formulation. These parameters are found experimentally via cross validation and the implementation notes on how it was conducted are given here. General hyperparameters are also included that were not necessarily found via experimentation. They are the split of the dataset into training/testing, the texton hyperparameter for the filter size, and the application map between filters/imagery. The reasonings for the selections are also provided. Evaluation methods that are common regardless of the experiment follows. These are fundamental metrics and plots such as the confusion matrix, and F1-Weighted Average vs. class index. This includes an explanation of the utility of such metrics. The experiments for K-convergence and % data convergence are detailed along with the methods to evaluate the results.

Chapter 6 Results & Conclusions discusses the two experiments conducted, K-convergence and % data convergence. Plots and tables described in chapter 4 Implementation are provided in the analysis. Plots of the classifier performance on the test set are the primary source of information, showing the performance as these parameters are swept. Plots of the number of samples per class for the test and training sets provide insight into the class imbalance

problem where the samples are not evenly distributed. The confusion matrices are standard for viewing misclassification between classes. These devices, plus $F1$ vs each class, can be used to show which classes are likely to be confused with other classes and the success of the classifier. Significant analysis is placed on whether the convergence plot matched that found in literature and possible explanations of this result. Mesh plots, or gradient plots, are used to show the cross validation stage results. The purpose being to maximize the $F1_W$ score. Training time is also plotted to give a reasonable idea of the time required to test and develop the system. The sections each end with example predicted imagery to show the final system. Frames were chosen for demonstrations of particular class disambiguations discussed here.

Chapter 7 Future Work discusses items that are promising avenues for the continuation of this system. The items are often brief but imagery or a more lengthy discussion is included were appropriate. Future work includes improvements to the hardware, additional variables, and the strategies used. Most notably there are key deficiencies in the hardware that if addressed successfully can greatly improve the work flow in data collection and greatly improve the quality of the data.

It is also worth noting here some strategies used to assist the reader. First, the the *hyperref* LaTeX package was used to color code in text references and citations as well as urls in the bibliography. In text references are encoded by the spectral color red, citations are encoded by the hue magenta, and urls are in the spectral color blue. This combination should assist users who have Dueteranopia or Protanopia. The encoding may not be as stark for Tritanopia, but the citations will always be numbers within square brackets [ ] as opposed to the references which do not have this prefix/suffix. These references and citations can be clicked to direct the Portable Document Format (PDF) GUI to the referenced item. Second, tiling is used often where multiple images are shown. In these cases the caption communicates the image order. MATLAB notation for matrices are often used: commas for horizontal delimiters and semicolons for the next line.

Chapter 2

Literature Review

This section addresses works that have used multispectral and polarimetric data particularly for material classification in unmanned ground vehicle settings. Texture and textons are also addressed but are not necessarily related to multispectral or polarimetric data. The section concludes with a brief discussion on practical concerns for material and object classification given similar studies in literature. Namely the data requirement, annotation strategies, and class selection are discussed.

2.1 Multispectral and Polarimetric Data in Perception

2.1.1 Mud Detection

Rankin and Matthies have explored mud detection with multispectral imagery in unmanned ground vehicle environments[4][9] of which can be a common and costly hazard for ground vehicles to resolve. Bands were analyzed within the optical light, optical polarimetric, NIR, SWIR, Mid Wave Infrared (MWIR), and LWIR spectrum. Each spectrum was analyzed separately, i.e. cameras were not spatially registered between one another after capture for the purpose of combining intensities across spectrum Although the study was constrained to "mud detection during daytime under ideal conditions", e.g. "isolated wet soil surrounded by dry soil", many practical insights are detailed according to the following categories: color and hue analysis, the soil line, SWIR, thermal infrared, polarimetric, and stereo information.

Mud Detection with Color and Hue Analysis

As explained in 1.3.4 Color Space, the color of a pixel can be separated from the intensity. This is important when dealing with dark objects, such as mud, that can be confused with

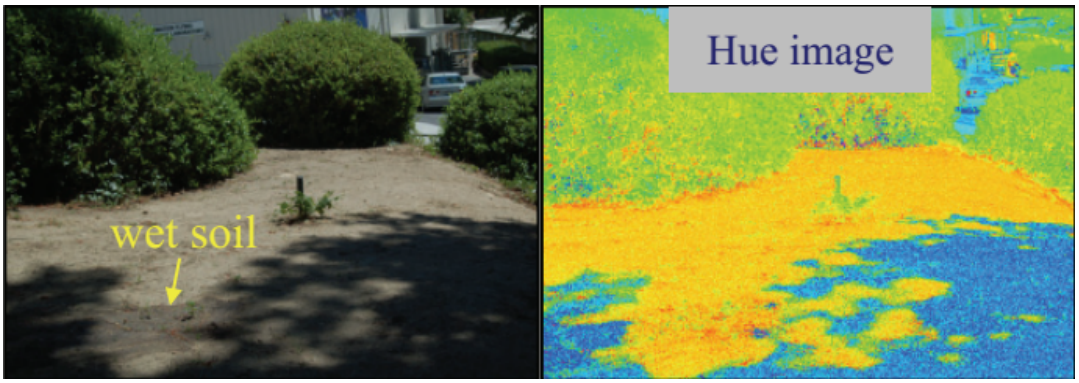


Figure 2.1.1: Comparison of dry/wet ground in the Hue representation of the VIS spectrum. The wet soil is slightly darker, and shadows contain more blue light than the surroundings[9]. L. H., R. Arturo L, and Matthies, "PASSIVE SENSOR EVALUATION FOR UNMANNED GROUND VEHICLE MUD DETECTON," Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, Tech. Rep., 2009. Used under fair use, 2015.

shadowed areas. Rankin and Matthies explain that because a shadow is illuminated indirectly "blue light tends to scatter more relative to green light"[9]. Therefore, mud has a characteristic of a lower intensity and a similar hue compared to the surroundings, while shadows have a lower intensity and a hue closer blue. This is shown in figure 2.1.1, where the wet soil is darker in the RGB image and similar in hue to its directly illuminated surroundings. The shadow areas show a hue closer to blue. Rankin and Mattheis suggest that hue should be useful differentiating mud from shadows but may not be when identifying mud in shadows. It is likely that shadowed areas should be treated separately from the material/object that the shadow covers given this insight. Namin and Pertersson has done so for the materials of grass and road[1]. The classes were included "as they were readily marked up" but not others which where not as easy to manually label[1]. Figure 2.1.1 is a good example of this as well. The shadow has many holes which will make the hand labeling process more difficult. Additionally, where does the shadow end and begin? The edges are not sharp. This is discussed further in 2.3.2 The Annotation Process.

Mud Detection with the Soil Line

Rankin and Mattheis stated that "it is well known that LandSat multi-spectral data on bare soil falls on a line (referred to as the soil line)" [9]. Indeed, the soil line is a linear relationship between the red and near infrared band, that can vary with soil moisture[56][57]. Rankin and Mattheis questioned if the soil line used for satellite imagery was valid for viewpoints on the ground. Their tests suggested that yes, the soil line is still valid for unmanned ground vehicle viewpoints and a lower moisture content results in a higher intensity of both bands, with shadowed soil darker than both wet and dry soil. Distance Along the Soil Line (DASL) and Normal Distance To the Soil Line (DTSL) can be calculated if the slope and y-intercept is

known[9] and serve as important features for mud/soil classification. The crux to this analysis is whether or not the soil line parameters are known. Rankin and Mattheis were able produce the soil line in their short experiment by plotting the intensities for the classes of soil, wet soil, and shadowed soil, and subsequently finding a best fit line to their hand labeled data. Soil line parameters vary with soil type such that it "should be defined for each soil" and "no universal soil line [exists]"[57]. Applying the DASL and DTSL in unmanned ground vehicle perspectives is therefore difficult and requires a priori knowledge. However, it does confirm that a pattern exists and the ratio between NIR & R is significant. This situation exhibits two crucial criteria for being a good candidate for machine learning: a pattern exists and we cannot pin it down mathematically[33]. It will be important to included NIR & R ratios as features in a future classifier for classes involving mud and dirt. Vegetation also exhibits patterns with respect to NIR & R light. Many vegetation indices and soil-line related features can be pursued in a parallel manner[57][3][58][59][60]. See 2.1.3 Vegetation Detection for further discussion.

Mud Detection with SWIR Imagery

Water absorbs NIR and SWIR light, which prompted Rankin and Mattheis to explore the feasibility of using SWIR imaging to measure soil moisture content[9]. They concluded that the targets of dry/wet soil, water, vegetation, and rocks do exhibit patterns but there are many confounding variables that complicate the issue. Measuring soil moisture content rather than differentiation between dry/wet for instance is likely not possible because while "SWIR intensity values decrease with increasing soil moisture content...the intensity differences can be visually barely noticeable"[9]. As the moisture content increases to the point that standing water is formed, the intensities decrease unless something other than the sky is reflected. The findings of the Mechatronics Lab matches the described patterns which are shown in figure 2.1.2. The bottom image pair shows reflections of clouds, blue sky, and vegetation. The clouds and vegetation are bright compared to the reflection of the blue sky. The middle image pair show a puddle that is reflecting blue sky and is dark. The vegetation and dry soil/gravel surrounding the puddle are both bright and difficult to distinguish from one another. Running water shown in the top image pair is dark in the SWIR image and no reflection in the RGB image is present.

Rankin and Mattheis conclude that other dark regions such as shadows, dead vegetation, snow, and water will need to be disambiguated from mud for successful detection and suggest color plus stereo information as likely solutions. Shadows are indeed stark in most cases. Generally speaking most green vegetation seen by the Mechatronics Lab has however been of similar intensity to dead vegetation. Figure 2.1.3 shows significant dead vegetation in the middle and lower image pairs, the difference between dead and green vegetation is very subtle and hard to detect visually. Also notice that bark for trees is dark in the SWIR but brush and thin branches are bright. Snow has also generally been bright in the SWIR band rather than dark, seen in the top image pair. Tracks that have exposed the ground and snow

with mud are darker than the surrounding clean snow. The shadows shown in the middle image pair are created by clumps of grass, and are indeed very stark in contrast to the surroundings in the SWIR band. Glass, some plastics, and some rocks have been also seen by the Mechatronics Lab to absorb SWIR light, but are not shown in the figures. The reason for these discrepancies between [9] and the Mechatronics Lab regarding dead vegetation and snow are not known at this time. It is possible that Rankin and Mattheis were referring to a SWIR camera that was sensitive to a different section of the SWIR band. Two SWIR cameras were used by Rankin and Mattheis: one from "900-1700nm", the same band used by the Mechatronics Lab, and one from "1460-1625nm"[9]. Additionally, the data presented in figure 2.1.3 and 2.1.2 represent only 2 days of data collection and are introduced as an exploratory representation rather than a generalized one. Special attention for the SWIR band in the future should be paid to shadows, snow, bark, and reflections.

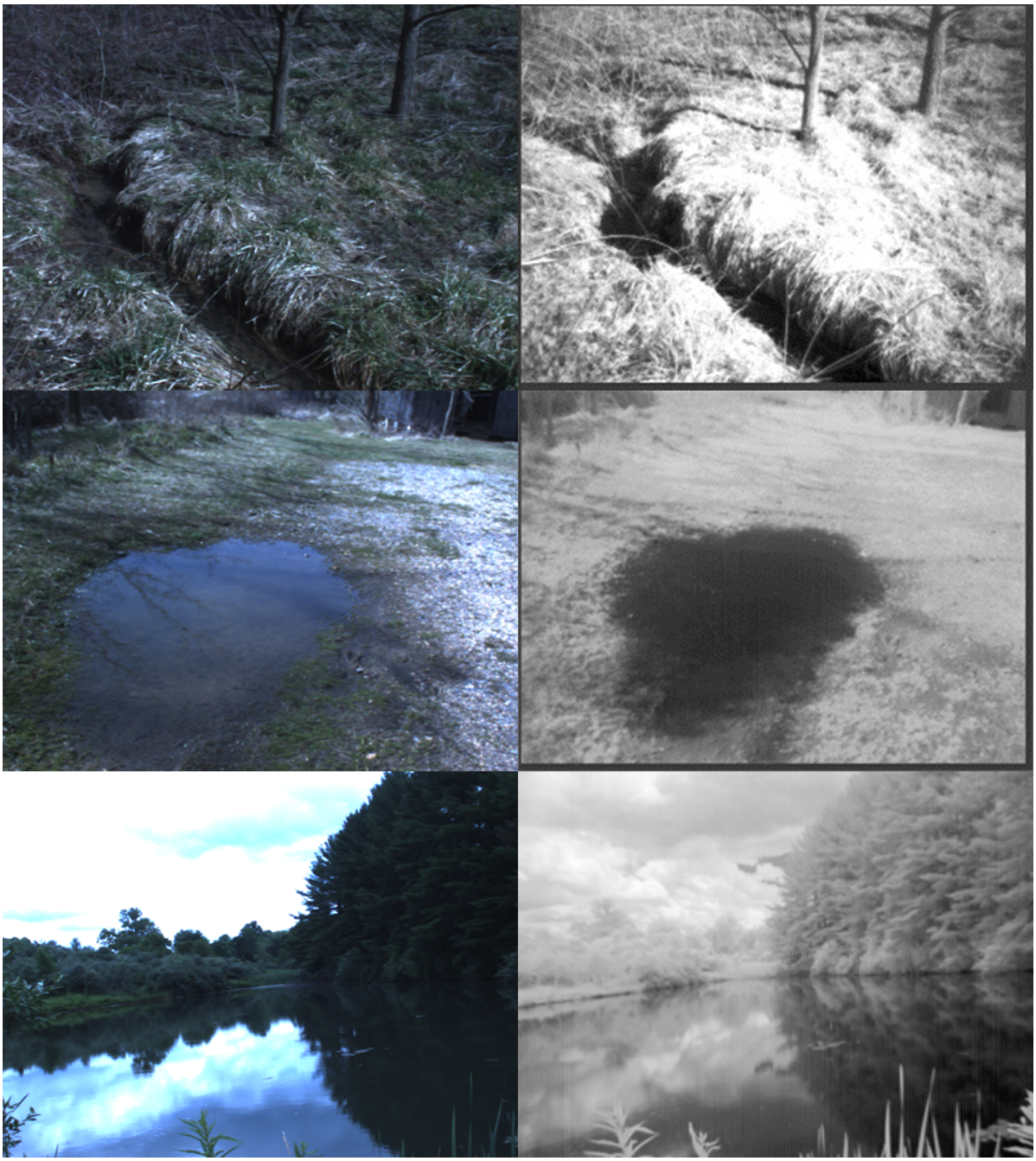


Figure 2.1.2: VIS and SWIR imagery showing water flowing, standing, reflecting the sky and clouds. Left column: visual light. Right column: SWIR $(0.9 - 1.7\mu m)$.

Figure 2.1.3: VIS and SWIR imagery showing confounding variables with snow, dead vegetation, and shadows. Left column: VIS. Right column: SWIR $0.9 - 1.7\mu m$. The top image pair was captured after snow in February. The middle and bottom image pairs were captured in March.

Mud Detection with Thermal Imagery

Rankin and Mattheis conducted exploratory tests to determine if mud detection is feasible using LWIR $7-14\mu m$ and/or MWIR $3.6-5\mu m$[9]. Temperatures and photos were recorded of mud, air, and dry soil during an all day test, summed up in figure 2.1.4. See [9] for the full temperature profile.

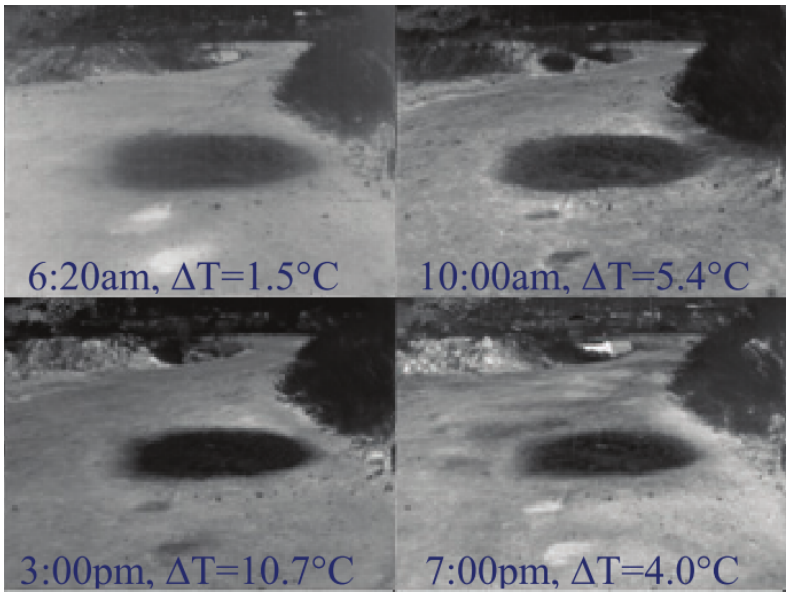


Figure 2.1.4: Mud in the LWIR band from an all day measurement showing the temperature difference between the mud and dry soil [9]. L. H., R. Arturo L, and Matthies, "PASSIVE SENSOR EVALUATION FOR UNMANNED GROUND VEHICLE MUD DETECTON," Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, Tech. Rep., 2009. Used under fair use, 2015.

The minimum temperature difference was 1.5° C, which was enough to provide a significant visual distinction. It was concluded that mud classification should be feasible, again identifying targets with similar low intensity signatures that could be confused with mud: shadow, snow, vegetation, and water. Figure 2.1.5 shows imagery from the Mechatronics Lab with these targets.

Figure 2.1.5: Examples of low intensity targets in the LWIR band, captured from a Xenics GOBI2689 ($8 - 14\mu m$) [10] from the Mechatronics Lab. Left to right: VIS, LWIR, SWIR. Top to bottom: snow, shadow, water.

Rankin and Mattheis suggest using color imagery to distinguish mud from snow, as snow is bright in the VIS band. Indeed the top row of figure 2.1.5 shows a thick and thin region of snow that is distinct in the VIS band but not in the LWIR. A region of mud is distinct in both bands in this case however.Rankin and Mattheis also recommend distance measurements to distinguish vegetation from the ground. Vegetation is generally cooler than bare ground, as seen in the middle row.

They suggest using stereo data and color data to distinguish reflections. The bottom row is an example of such. One can see that the LWIR image does not have much texture if any and is further blurred in the reflection. LWIR to LWIR stereo information will likely produce a sparse stereo correspondence due to the lack of texture. Applying a VIS to VIS stereo correspondence for labeling reflections in the LWIR would require LWIR to VIS registration. They also posit that thermal imagery will not be useful when the entire ground has been covered with water as there will no longer be a significant contrast. Identifying the areas

based on relative moister content would also be difficult due to the low contrast. Finally, Rankin and Mattheis show that in some cases mud that is occluded by leaf and other debris is identifiable in the MWIR imagery, shown in in figure 2.1.6. A similar phenomena observed in the SWIR and LWIR band by the Mechatronics Lab involving cut grass and patches that incompletely cover the ground, seen in figure 2.1.7. A similar pattern is detected in tracks from cars but with less contrast.

LWIR is a desirable band for imaging especially if temperature information is available rather than relative heat. Water, vegetation, and wet/dry soil exhibit patterns that could be exploited. Registering the LWIR band will be a crucial step in applying the LWIR data towards a classifier, especially given the lack of texture in the LWIR image.

Figure 2.1.6: Example of mud occluded by pine needles, identifiable in the MWIR image (right)[9]. L. H., R. Arturo L, and Matthies, "PASSIVE SENSOR EVALUATION FOR UNMANNED GROUND VEHICLE MUD DETECTON," Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, Tech. Rep., 2009. Used under fair use, 2015.

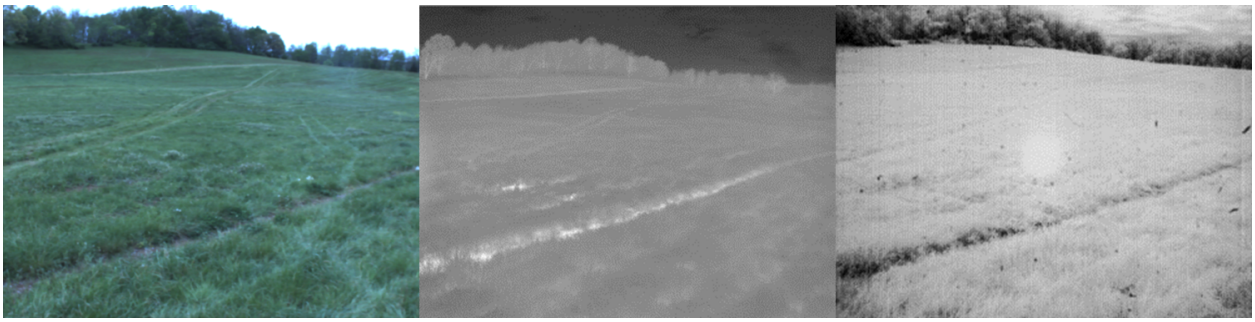


Figure 2.1.7: Ruts in a grass field shown left to right: VIS, LWIR ($8-14\mu m$), SWIR ($.9-1.7\mu m$).

Mud Detection with Polarimetric Imagery

Light becomes horizontally polarized if reflected off of smooth water[6] or mud[9]. The degree of linearly polarization(DOLP) or polarization contrast can be calculated to detect this change in polarization. The DOLP varies from 0, no polarization present, or 1, completely polarized. It can be calculated from either the Stokes parameters [6] or through pixel intensities with relative filter orientations of 0°, 45°, 90° [9]. Rankin and Mattheis simplified this by assuming that one of their filters is aligned roughly with the direction of polarization for mud reflections, and used the normalized absolute difference between the horizontal filter (0°) with maximum transmission or reflections and the vertical filter (90°) with minimum transmission. Therefore areas of the image that have a higher DOLP or polarization contrast are good candidates for mud/water.

Rankin and Mattheis conducted two tests on feasibility given that the polarization of the reflected light off of targets can change "based on sky conditions, the sun position, and the sensor orientation"[9]. The first test calculated the polarization contrast over an all day collection for a static camera, and they concluded that mud is separable from dry soil in the polarization contrast image irrespective of the sensor position relative to the sun. The second test calculated DOLP of a target in 4 cardinal directions over a 20 minute period. Again, the mud was separable from the dry soil as it consistently had a higher DOLP in all directions.

Rankin and Mattheis identify two situations that can confuse mud detection via polarization: mud in shadows and reflections on water. The shadowed situation can be seen in figure 2.1.8, where both polarization contrast and DOLP was measured to be visually similar for mud in and out of shadows. Standing water was also found to have a similar signal as mud. Rankin and Mattheis suggest using color and stereo range data to distinguish water from mud candidates similarly to the thermal solutions.

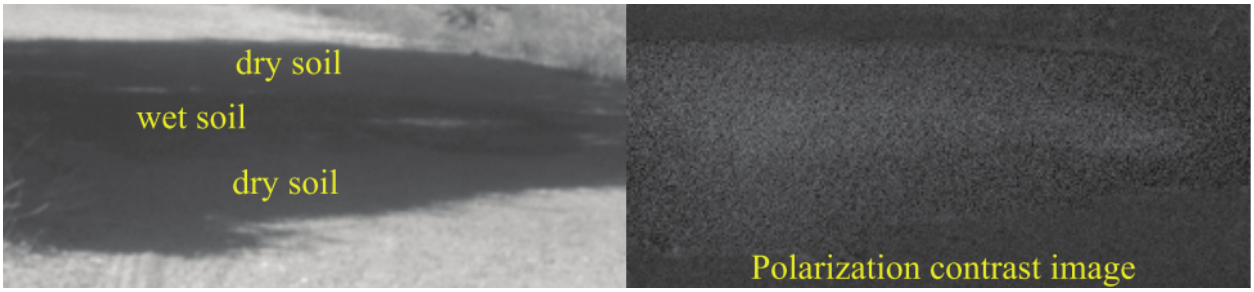


Figure 2.1.8: Wet and dry soil in shadow, in the reflected non-filtered light (left) and the polarization contrast image (right). The shadow makes mud more difficult to segment from dry soil using polarization contrast [9]. L. H., R. Arturo L, and Matthies, "PASSIVE SENSOR EVALUATION FOR UNMANNED GROUND VEHICLE MUD DETECTON," Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, Tech. Rep., 2009. Used under fair use, 2015.

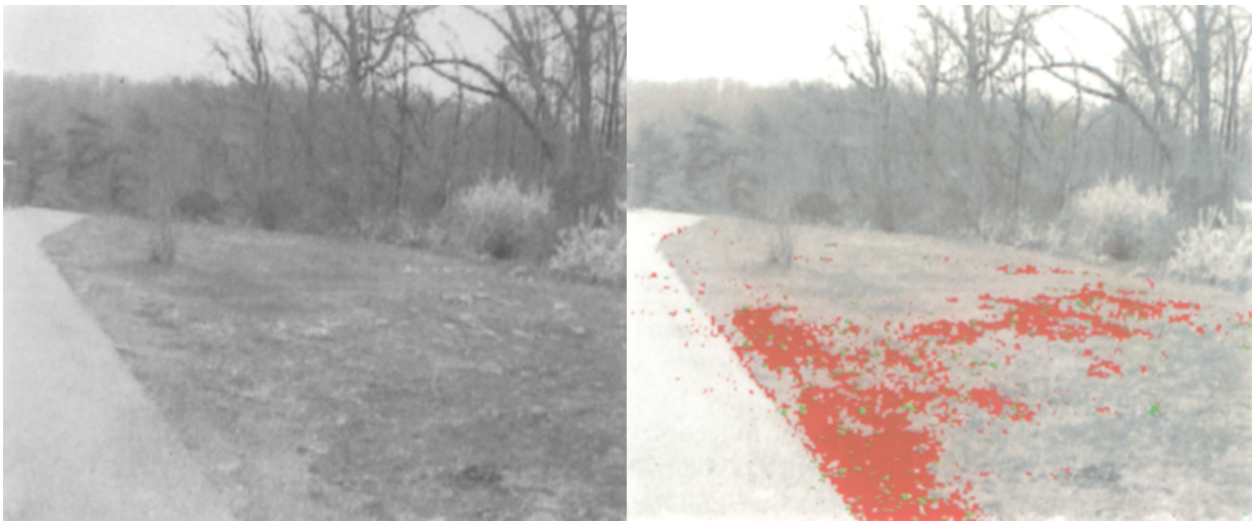


Figure 2.1.9: Dirt segmented via phase and DoP. Original image (left), segmented image (right). For pixels with DoP $\geq$ 5%: red when $-20° \leq \theta \leq +20°$, green when else [5]. L. B. Wolff, "Polarization vision: a new sensory approach to image understanding, "Image and Vision Computing, vol. 15, no. 2, pp. 81-93, Feb. 1997. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0262885696011237. Used under fair use, 2015.

The phase of polarization is also an important metric. The phase from reflections "directly reveals the orientation of the [intersection of the surface normal to the imaging plane]"[5]. This is demonstrated in figure 2.1.9, where pixels are segmented as long as there is at least a 5% level of polarization, the ground is segmented as red when the phase was within $\pm 20°$ from the horizontal ($0°$) plane, and pixels "significantly deviant from horizontal"[5]. The green pixels are difficult to perceive, but a significant area is seen as red. The red indicates a relatively flat surface while green indicates a protrusion, in this case it likely represented gravel that is pointing out at an angle. Wolff notes that dirt and rock polarize light after reflection much more than vegetation does. A single image set is shown to explore this phenomena. Figure 2.1.10 shows degree of polarization remapped from $0 - 1$ to $0 - 255$ and phase remapped from $0 - 180°$ to $0 - 255$. The targets present are asphalt in and out of shadow, grass & foliage, and tree trunks. Reflections off of the asphalt are indeed closer to 0 in the phase image (near black 0, or white 255). However this is not consistent, there are patches of asphalt that are in full sun but have a phase closer to $90°$ (gray). The areas of asphalt in shadow have a phase closer to $0°$ and have a high DoP. The vegetation has a high DoP when in shadow but mostly a low DoP out of shadow. It is also curious that the bark of the trees have a high DoP and a phase close to $0°$.

DoP and phase will be significant features for image classification. Questions to explore include: What is the appearence of vegetation with respect to DoP, phase and shadows? Will flat reflective surfaces have a phase close to zero? Will these features influence labeling of man-made materials since those are typically smoother than objects found in nature?

Figure 2.1.10: DoLP and phase for a suburban scene. Clockwise starting at top left: VIS, VIS + NIR I_0 polarization, phase remapped 0-180° to 0-255, DoLP remapped 0-1 to 0-255.

2.1.2 Water and Sky Detection

Standing water is a difficult target to segment using pixel intensities due to reflections off of the water surface. Figure 2.1.11 shows that these reflections are present in all the bands used by this study. While some intensity and texture is lost upon reflection, most is preserved when the water is still, top image of 2.1.11. More texture is lost when the water surface becomes disturbed, seen in the bottom image of 2.1.11. This section addresses techniques for resolving this confusion between the targets and reflections.

Figure 2.1.11: Examples of reflections off water in raw imagery. Left to right: VIS, NIR, SWIR, LWIR. Note that reflections of cloud, clear sky, and the vegetation is present in all bands. Texture is lost where the water surface is disturbed, seen in the bottom frame.

Polarimetric

Light becomes partially polarized or polarized upon reflection as explained in 1.3.2 Polarization by Reflection. "Since water is a fluid and therefore oriented parallel to gravity, this provides a physically-based methodology for segmenting water"[5]. The two features that polarimetric imaging provide are DoP and phase.

$$BrewsterAngle \quad \theta_B = \arctan n_r/n_i \tag{2.1}$$

where

θ_B is the angle that the reflected light is linearly polarized
n_i is index of refraction of the material for the incident light
n_r is index of refraction of the material transmitted light

Unpolarized incident light will be perfectly polarized at the Brewster Angle and partially polarized otherwise, shown by equation 2.1[29] and figure 1.3.5. Given that the index of refraction for air is 1.000277 and water is 1.333333[29] the $\theta_B = 53°$ for reflections off water for this study. The camera than should be pointed downward at 53° from the horizontal plane perpendicular to gravity in order to sense the maximum of this effect. However, if the sun is not positioned relative to the water by 53° then the majority of incident light will be from scattering and reflection off of objects. The sensitivity of these polarization measurements to the sun does not appear to be prohibitive however. Photos taken of by Umansky showed that water and grass were still separable given different viewing angles and whether or not the sun was occluded by clouds[6]. Preliminary results of this study agrees with that observation.

Reflections off of water will have the feature of partial to complete polarization dependent on the angle of observation and the angle of the sun relative to the target and observer.

The polarization phase, or the orientation of polarization, is the direction of the electric field vector as explained in 1.3.2. It is perpendicular to the plane of incidence. Light in the sky is partially polarized due to Rayleigh scattering and is oriented almost vertically[5]. Clouds scatter light in an effect called Mie scattering[29] and the reflected light is unpolarized[5]. A scene containing water reflections will show reflections of the sky and clouds. The reflections of clouds will have a phase "oriented horizontally", while the reflections of sky with an "approximate orientation range of $40° - 60°$"[5]. Wolff[5] has excellent images of a beach showing this effect with phase remapped to hue values. Figure 2.1.12 shows similar a similar effect with preliminary results of this thesis, showing phase remapped from $0-180°$ to $0-255$.

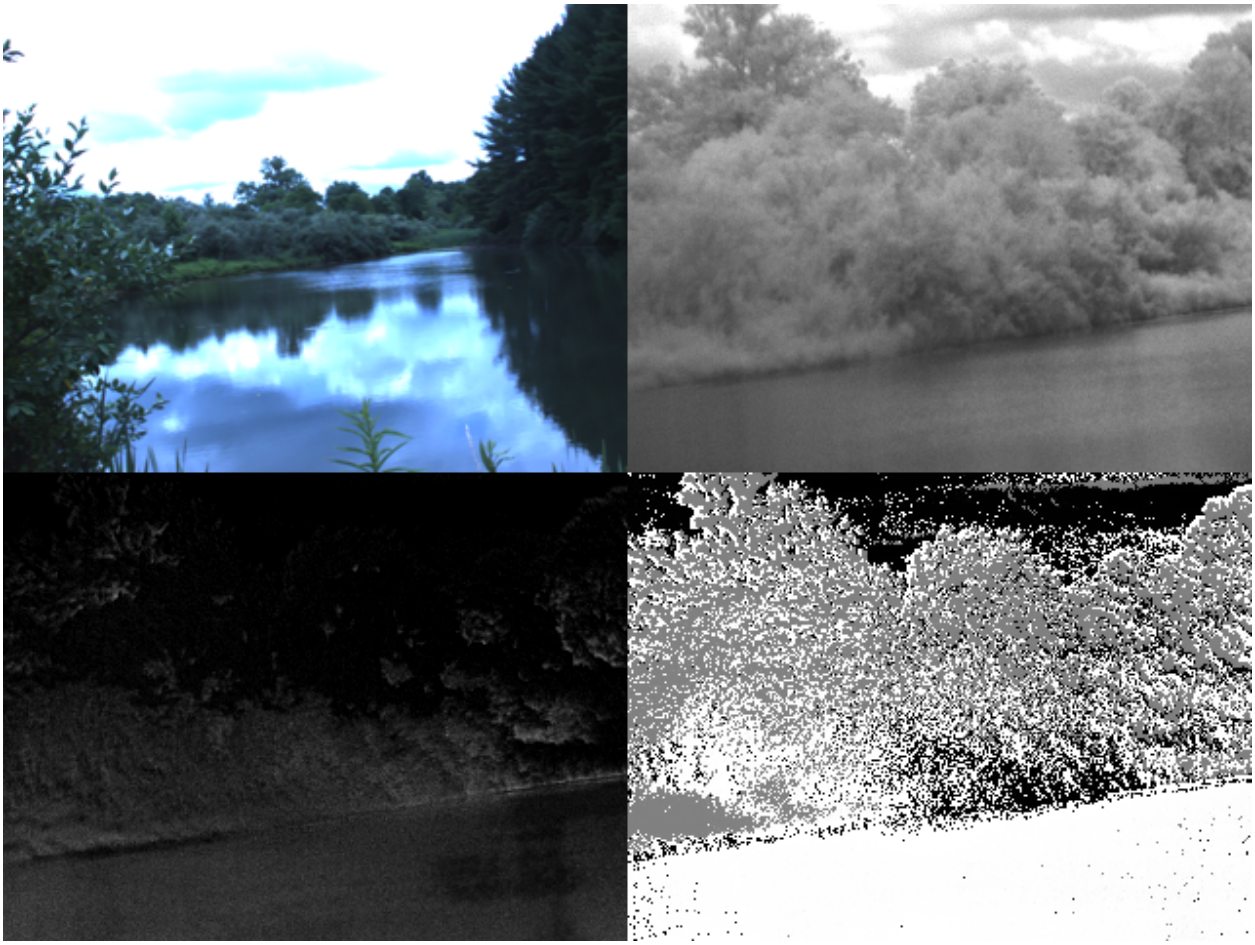


Figure 2.1.12: Example of water reflections in post processed imagery. Clockwise from top left: VIS, Horizontal POLAR, phase remapped from 0-180° to 0-255, DoLP remapped from 0-1 to 0-255.

The water reflections are white in the phase image signifying that the phase is close to 0°. Said reflections are of vegetation, which preliminary results of this thesis have shown to be predominately unpolarized with the exception of vegetation in shadows. The clouds in the sky are predominately close to zero in phase, and the small sections of sky appear to be gray, indicating a phase near 90°. The ability to collect data containing reflections of sky and clouds on water at the same time is complicated by the narrow FoV of the polarimetric camera using in this thesis.

Horizon & Sky Reflections

Rankin, Matthies, and Bellutta[11] also used a reflection feature to detect standing water in two separate strategies. The first relied on a prior segmentation of the sky within the image and a level camera platform (minimized roll). For each image column the bottom N sky pixels were sampled to form an average color and texture for the sky. The remaining pixels in each column were classified as reflections if the features were close to the average. This was effective for distant targets only and a level platform where gravity was aligned to the image columns. The second technique relied on knowing extrinsic camera parameters that make up its pose such as roll, which were polled from the test vehicle IMU. It exploited the fact that standing water acts as a mirror of which reflects a specific part of the sky, and named the reflection off of the water as the reflected ray, and the original point in the sky the direct ray. For each candidate water pixel the image coordinates of the direct ray were predicted using the downward angle that the camera was pointing and the direction of gravity. The candidate was labeled as a reflection if features of the candidate water pixel and the corresponding direct ray were sufficiently close. Similarly, this technique was not as effective at close range: at approximately 6 meters in front of the vehicle the image coordinates of the direct ray were outside the FoV and hence the comparison could not be performed. This was the case for a camera tilt of 10° downward from the horizontal, a camera height of $1.5m$, and a 48° vertical FoV. Once the direct ray pixel has passed the top of the camera FoV however, Rankin et al. points out that "the color coming out of the water body dominates and knowing the color of the sky reflecting on a candidate water pixel is of marginal use"[11].

One important assumption was that the ground was assumed to be horizontal and flat, most likely to calculate the distance along the ground from the sensor although it was not specified. Though this may not be a safe assumption for previous work of the Mechatronics Lab, this nonetheless illustrates two important features: 1) location of the horizon and direction of gravity 2) context of pixels oriented with respect to gravity.

Knowing the horizon location and orientation provided Ranking et al. the line with which to calculate reflections across. This could likely be applied to many other classes. For example, structures such as tree trunks, light poles, and buildings are oriented parallel to gravity, standing water is oriented perpendicular. Rock and dirt are much more likely to be below the horizon while vegetation and buildings are more likely to be above. Similarly, using the

horizon to look for cues across the reflection line is providing a form of contextual clues. Features describing shape could also be used in addition to intensities. Figure 2.1.12 is an example where the edge between th sky and the tree canopy has a shape that is reflected in the water reflection.

Saturation to Value Ratio

Rankin, Matthies, and Bellutta[11] identified features extracted from the VIS space singly and combined with pose information that were effective at detecting reflections on still bodies of water. Their analysis used the HSB color space, which is uncommon to the resources used in this study. The HSB space is considered the same as HSV where the *Brightness* term is used interchangeably with *Value*, the former being more common in Adobe Photoshop[48]. The HSV space is used here for consistency. Pose and distance data was used to estimate the horizon which enables one to detect water bodies "indirectly...by detecting reflections of the sky below the horizon"[11]. Rankin et al. identified the following features from the VIS imagery without the pose and distance data to be important: 1) *Saturation* to *Value* ratio 2) intensity variance 3) edge magnitude.The features exhibited phenomena specific to cloudy, overcast, and clear skies. Namely: 1) cloudy and clear skies have low S/V ratios 2) clear skies have low variance of intensities. Water in particular was identified to have low variance in S/V ratio between the edges of the pool.

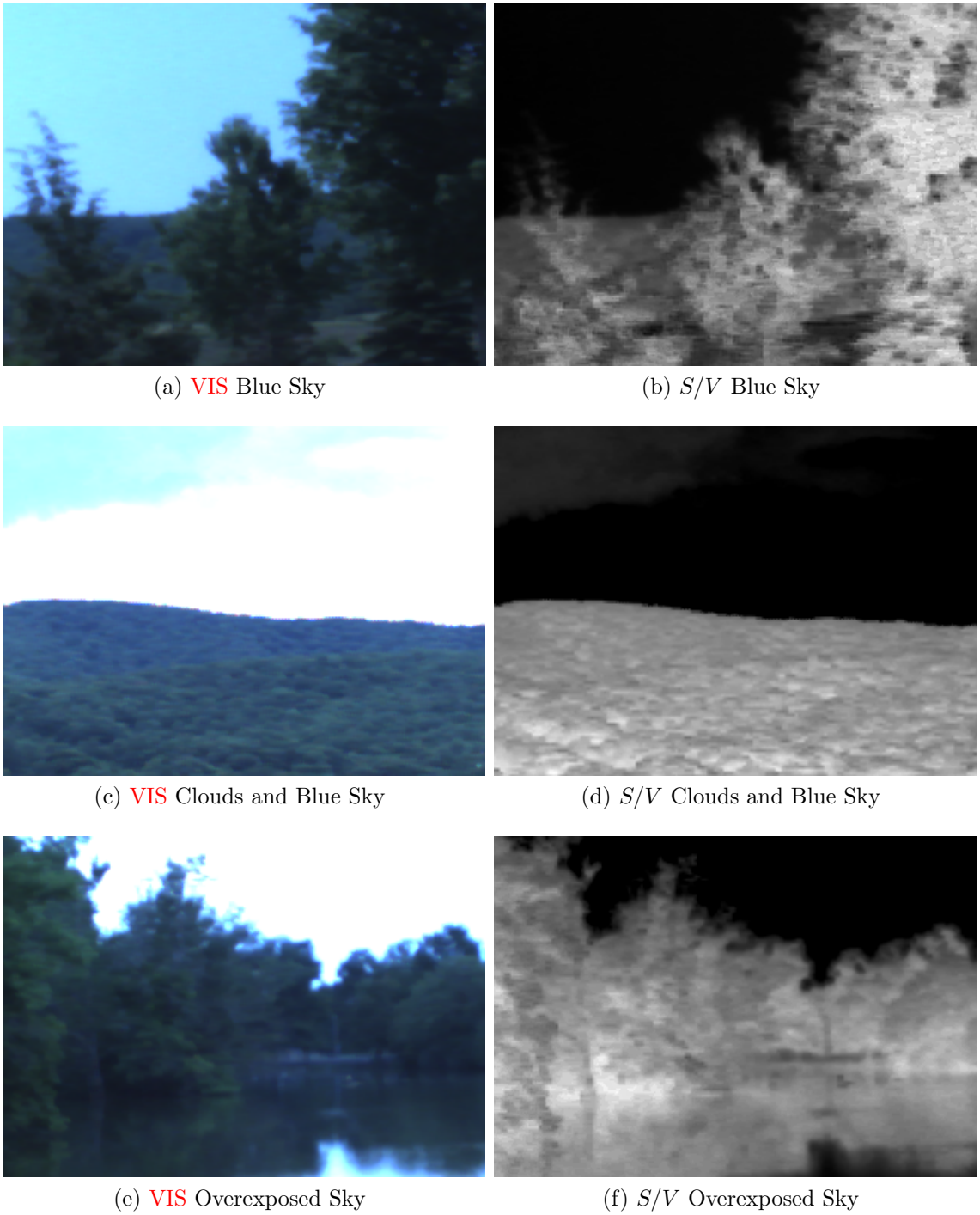
Initial tests of the S/V feature show promise for sky, reflection, and water detection in figure 2.1.13. Similarly to Rankin et al., the texture of the sky is very low, i.e. flat and with minimal variation. The S/V intensity is high for vegetation, and the texture is lower after reflection in the preliminary tests. Note that Rankin et al. describe their images where the sky is white, i.e. low saturation (color) and high value (intensity), as "Overcast" and here *overexposed* images are shown. This is because it is difficult to identify form the imagery alone if the weather was indeed overcast and exposed properly for the sky. It is much easier to identify overexposure for the sky which is shown by low saturation, high value, and a loss of texture on the boundaries of the skyline. See figure 2.1.13e for an example.

Edge information is addressed in 2.2 Texture & Textons and 4.2 Filter Implementation.

Moving Water

Disturbed water can confuse reflection detection as the ripples cause an irregular distortion, seen in figure 2.1.11. If reflections are distorted or removed, what features are applicable? Examples in literature show that polarimetry is still valid[5][6]. Both sources have shown that a moving stream still partially polarizes light upon reflection, and one of the two demonstrated that disturbed water on the ocean partially polarizes upon reflection.

The lack of reflections could be seen as a boon rather than a loss, no longer is intensity



(a) VIS Blue Sky

(b) S/V Blue Sky

(c) VIS Clouds and Blue Sky

(d) S/V Clouds and Blue Sky

(e) VIS Overexposed Sky

(f) S/V Overexposed Sky

Figure 2.1.13: Preliminary test of the *Saturation/Value* feature from [11]. The S/V frames are rescaled to U8 relative to the bounds of each image. The intensities are not necessarily comparable between frames as a result.

such a confusion feature.Figure 2.1.14 is an example where moving water is dark in NIR, LWIR, and SWIR. This matches with literature as NIR and SWIR are absorbed by water much more strongly than VIS[61]. SWIR ($\geq 1000nm$) is much more strongly absorbed by water than NIR ($865nm$)[62]. The water is dark in the LWIR band due to the temperature difference.

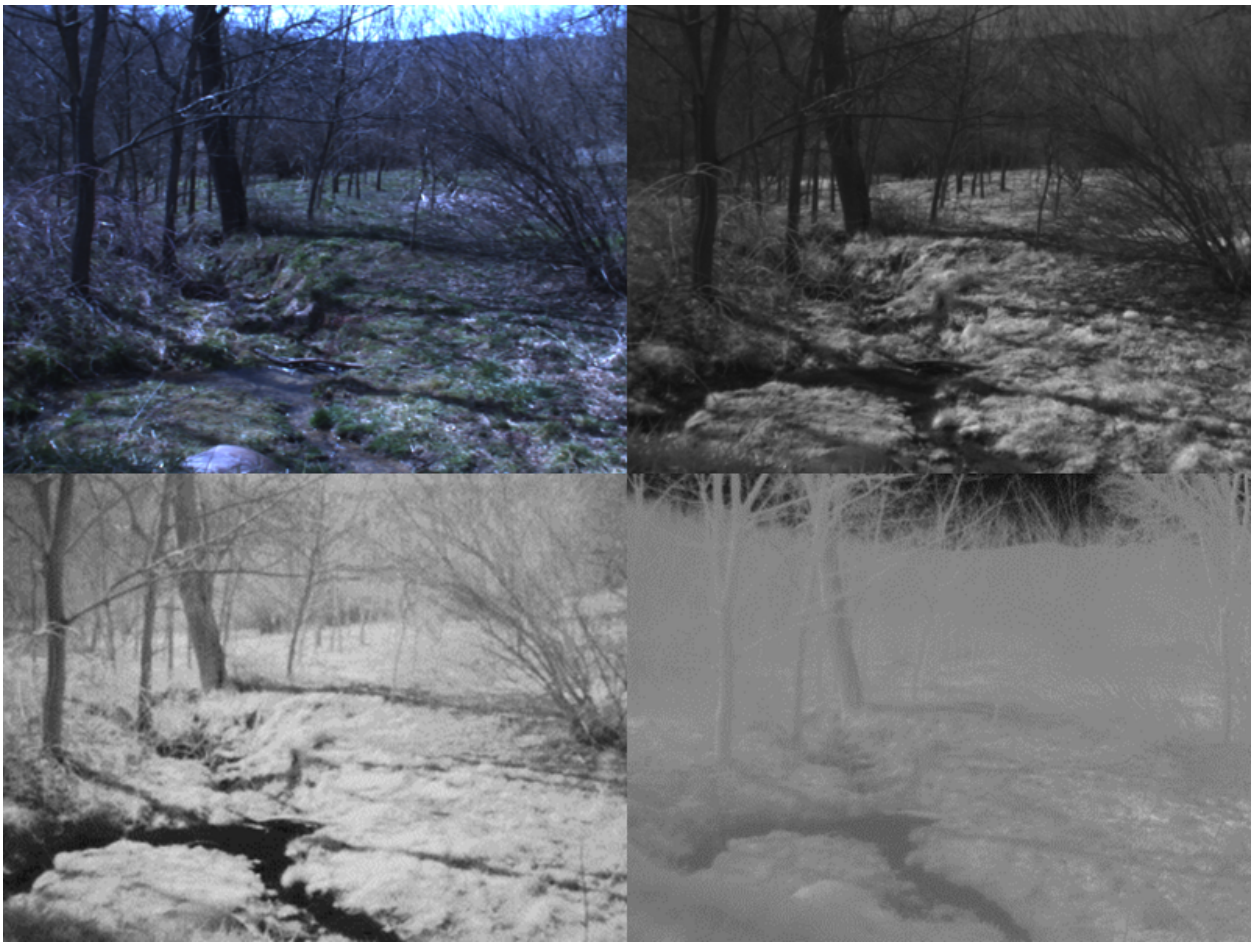


Figure 2.1.14: Moving water in raw imagery, from top left clockwise: VIS, NIR, LWIR, SWIR. Note that in the VIS image reflections are minimal and one can see through the water.

Figure 2.1.15: Example of shallow still water in raw imagery. From left to right VIS, NIR, SWIR.

Moving water in this passage has been used to describe any water surface that is not undisturbed. This passage is by no means comprehensive as there are many situations that have not been taken into account. The reflectance of NIR by turbid water for example r, i.e. water "with high sediment concentration", is significant for remote sensing applications[62]. Also, it has been observed that reflections are less significant in NIR and SWIR compared to VIS in shallow puddles, seen in 2.1.15.

Given these few examples and observations, this thesis will continue on the assumption that DoP is likely a useful feature for both still and moving water, and the classification of sky and reflections will benefit from features encoding context, variance, hue, DoP, phase.

Stereo Reflections

Rankin, Matthies, and Huertas explored using stereo range data to detect reflections on still water, while fusing the range data with other features such as texture[63]. The range portion of their study operated on the principle that "the range to a reflection roughly matches the range to the reflected object" where the reflection object will appear to be beneath the ground surface. The range image was scanned column-wise for drastic changes in distance starting from the top and working downward. The range image will exhibit a large first derivative of the distance with respect to the row index when the reflection meets the water boundary, i.e. the top edge of the puddle. As the column is scanned further down from the top of the image, a feature describing the lower edge of the puddle is computed. If the lower edge of the puddle is not visible the reflection is rejected. If the lower edge is visible a line is fitted from the lower edge to the measurements above it within the reflection. The column is thresholded to reject any lines with an incline $\geq$ 40°. The candidate reflection is also thresholded on other features; for example: reflections that start at an elevation higher than the vehicle $\geq$ $1m$ are rejected, candidate reflections that start $\geq$ $50m$ away are rejected. Zero disparity (no detectable distance) pixels in the stereo data are also fused in a similar manner that texture is fused with the reflection feature. These range reflection features create false positives when looking through porous objects such as branches. Rankin et al. mitigated this

with the requirement that the near edge of the puddle must be visible, and through fusion of the other intensity features.

Previous work in the Mechatronics Lab has observed a similar effect when detecting standing water through LIDAR: reflections off the surface can appear to be a large hole in the road. In accordance with the initial motivation of this thesis, it is recommended to supplement the LIDAR with imagery to combat this problem.

Normalized Difference Water Index (NDWI)

The Normalized Difference Water Index (NDWI) is a common feature used to detect bodies of water, soil moister, and the water content of vegetation, particularly in remote sensing imagery. NDWI was developed on the same principle as NDVI, in that there is one band that is reflected strongly and another that is absorbed by the target. The difference of the two bands is normalized by the sum producing an output between $[-1, ..., 1]$. The placement of the negative sign is not necessarily important by should be consistent. The bands chosen for NDWI vary depending on the imagery available and the goals of the study. The boundaries of the hyperspectral and multispectral bands for example can differ between satellites. The NDWIs found are summarized in equations 2.2[64],2.3[65],2.4[66],2.5[67], 2.6[68]. The band boundaries are printed in nm if such information was available. It is important to note that wavelength boundaries for light do not necessarily have one standard, [66] considers $1230-1250nm$ to be NIR but was shown as SWIR here for consistency. Therefore the bands of this system shown in figure 1.3.1 are reprinted in table 2.1 here.

$$NDWI = \frac{G - NIR}{G + NIR} \quad [64] \quad (2.2)$$

$$NDWI = \frac{NIR_{860} - SWIR_{1300-2500}}{NIR_{860} + SWIR_{1300-2500}} \quad [65] \quad (2.3)$$

$$NDWI = \frac{NIR_{850-890} - SWIR_{1230-1250}}{NIR_{850-890} + SWIR_{1230-1250}} \quad [66] \quad (2.4)$$

$$NDWI = \frac{R_{620-670} - SWIR_{2105-2155}}{R_{620-670} + SWIR_{2105-2155}} \quad [67][69] \quad (2.5)$$

$$NDWI = \frac{G_{\sim 510-\sim 600} - SWIR_{1500-1750}}{G_{\sim 510-\sim 600} + SWIR_{1500-1750}} \quad [68][70] \quad (2.6)$$

Table 2.1: Wavelength boundaries of the bands available to this study

Camera	Band	γ	Source
JAI AD-080GE	VIS	$400 - 650\ nm$	[22]
JAI AD-080GE	NIR	$760 - 1000\ nm$	[22]
Gobi-384	LWIR	$8000 - 14000\ nm$	[22]
SU320HX-1.7RT	SWIR	$900 - 1700\ nm$	[23]
Polarimetric	VIS & NIR	$420 - 1000\ nm$	[24]

Liu states that "Water reflects most radiance in [the] visible wavelength...and NIR is absorbed strongly by water but reflected strongly by...vegetation and soil"[64]. This led to the choice of equation 2.2 using G/NIR, of which it was important to the author to discriminate between water surfaces and other objects. Equation 2.3 was chosen with the goal of detecting droughts as it was seen to be "sensitive to vegetation, soil moister, and leaf water content"[65]. Equation 2.4 was chosen with the goal of detecting soil moisture content and because the channels used were less likely to be affected by cellulose. These considerations are not applicable to this study since the multispectral bands are larger in breadth than the hyperspectral bands of [66]. It is noted however that in general "liquid water absorption in the 1500 nm - 2500 nm is significantly stronger [than 900 nm - 1300 nm]"[66]. Reflectance off, presumably standing, water is also shown to be nearly 1 between 400-900 nm, while it decreases strongly to 0.2 above 1400 nm[66]. Equation 2.5 was chosen with the goal of detecting droughts, and used NIR and a section of SWIR outside the range available to this study which was said to be the "longest [band] in the reflective solar bands"[67]. These bands were chosen as their use enable the authors to exclude noise from "clouds, cloud shadows, fires, in-land lakes, and show"[67]. Equation 2.6 as chosen with the goal of detecting water bodies with lower false positives due to mountain shadows compared to their alternatives. The use of the "mid-infrared band [1500 - 1750 nm] enhances the contrast between water and building, and greatly reduces the influence of building information on water"[68].

2.1.3 Vegetation Detection

A standard feature used for vegetation detection is the Normalized Difference Vegetation Index (NDVI) is shown in 2.7, which uses a ratio of NIR and red light. One of the earlier uses was for remote sensing analysis by Rouse, Haas, Schell, Deering, and Harlan in 1974[71] using $NIR = [8000, ..., 1100]nm$ and $red = [600, ..., 700]nm$. The feature is a chlorophyll-like response and can be simplified to 2.8 the Infrared Percentage Vegetation Index (IPVI)[72]. The two are functionally the same to a learner which will accommodate the scaling: NDVI is bounded by $[-1, ..., 1]$ while Infrared Percentage Vegetation Index (IPVI) by $[0, ..., 1]$. These bands are combined as "green vegetation" significantly reflects NIR light and absorbs red[55]. The high absorption of red is termed the "red edge" particularly in hyperspectral remote sensing where there exists a high first derivative of reflectance with respect to wavelength at

wavelengths $\lambda = [680, ..., 750]nm$[55]. This is due to "high internal leaf scattering" of NIR and "chorophyll absorption" of red and blue[55]. Bradley, Thayer, Stentz, and Rander[3] explain that water absorbs light with $\lambda > 1400nm$ highly, chlorophyll reflects red and blue light highly, and the difference in the refractive indices between water and air causes light at $\lambda = [800, ..., 1400]nm$ to be reflected and scattered highly off of vegetation. These phenomena have been used to produce a large number of vegetation features using the VIS alone or in combination with NIR.

$$NDVI = \frac{NIR - R}{NIR + R} \tag{2.7}$$

$$NDVI + 1 = \frac{NIR - R}{NIR + R} + \frac{NIR + R}{NIR + R}$$

$$NDVI + 1 = \frac{NIR * 2}{NIR + R}$$

$$IPVI = \frac{NDVI + 1}{2} = \frac{NIR}{NIR + R} \tag{2.8}$$

The Ratio Vegetation Index (RVI), equation 2.9[73], was an early vegetation feature developed in 1969 combining red and NIR. As Bradley et al. note the bounds of $[0, ..., +\infty]$ are difficult to work with, leading to the use of NDVI which is "functionally equivalent"[3].

$$RVI = NIR/R \tag{2.9}$$

The Excess Green Vegetation Index (ExG) in equation 2.10 was developed to "[distinguish] weeds from a non-plant background"[74], and uses VIS data without NIR. A constrained form Modified Excess Green Vegetation Index (MExG) equation 2.12[75], was used to correct for "camera effects" such as "...saturation of pixel values on the border of leafs"[60]. Further indices explored by Woebbecke were $r-b$, $g-b$, $(g-b)/(r-g)$[74]. Meyer and Neto analyzed the above indices, including Excess Red Vegetation Index (ExR) 2.11[76] and Normalized Difference Index (NDI)2.13[77], and found ExG and ExG-ExR to be in general more effective that NDI.

$$ExG = 2 * g - r - b \tag{2.10}$$

$$ExR = 1.4 * r - g \tag{2.11}$$

where

$$r = \frac{R}{R+G+B} \qquad g = \frac{G}{R+G+B} \qquad b = \frac{B}{R+G+B} \qquad R,G,B \in [0,...,1]$$

$$MExG = \begin{cases} ExG & \text{if } (G \geq R) \;\&\&\; (G \geq B) \;\&\&\; (G \geq 120/255) \\ 0 & \text{else} \end{cases} \tag{2.12}$$

$$NDI = \frac{G-R}{G+R} \tag{2.13}$$

2.2 Texture & Textons

Texture is a pixel level feature describing the local intensity pattern. The concept of a texture is often represented by image swatches of individual materials such as grass, brick walls, gravel, baskets of fruit, etc. Each material has a pattern that if reproduced, could extend the resolution of the image to appear larger. Material classification often uses data in the form of these swatches and often uses texture as a primary feature. The textural section of the USC Signal and Image Processing Institute[78] for example includes image swatches and mosaicked frames of grass, bark, cloth, sand, and so on. Textons are the unsupervised classifications of texture measures, which have been grouped to decrease the dimensionality of the feature list. Texture and textons are discussed here given the use of texture in material classification[54] and the success of textons in image segmentation[79] and object detection[14]

Many strategies exist for the production of textural features and descriptors. Those mentioned here are the Co-occurrence Matrix, Laws Texture Energy Measures, and the Gabor filter set.

Co-occurrence matrices encode the relationship between two pixels in an image. Using the notation of [43], a co-occurrence matrix C is 2D and square where the row and column indices represent the bounds of the pixel intensities, e.g. $[0,...,255]$ for *uchar* representation. The value $C[i,j]$ counts the number of times a pixel value of i "co-occurs" with the pixel value j in one specific spatial relationship $d = [dr, dc]$[43]. The vector $d = [dr, dc]$ here is the displacement in rows, dr, and columns, dc, such that the pixel pair in image I is $i = I[r,c]$, $j = I[r+rc, +cd]$. The co-occurrence matrix C is incremented at the location $C[i,j]$. The matrix is specific to one displacement vector and so is denoted by C_d. This is detailed in algorithm 1. The co-occurrence matrix is normalized shown by equation 2.14, which transforms the entries to probabilities[43]. The matrix is also transformed by equation 2.15 which "groups pairs of symmetric adjacencies"[43]. Features are created using the co-occurrence matrix, such as equations 2.16 to 2.20, as the entries themselves are not suitable

[43]. The co-occurrence matrix is also termed the *gray scale dependency matrix*, and the displacement vector d can be described in polar coordinates (r, θ)[46].

Algorithm 1 Population of a co-occurrence matrix C_d from image I.

Input: I, d=[dr,dc]

1: unsigned int i, j, k, l
2: C_d = zeros matrix, size $(I.rows, I.cols)$
3: **for all** $k < I.rows$ && $l < I.cols$ **do** ▷ zero indexing
4: **if** $(k + dr) \geq I.rows$ || $(m + dc) \geq I.cols$ **then** continue
5: **end if**
6: $i \leftarrow I[k, l]$
7: $j \leftarrow I[k + dr, l + dc]$
8: **if** $i == j$ **then** $++C_d[i, j]$
9: **end if**
10: **end for**
11: **return** C_d

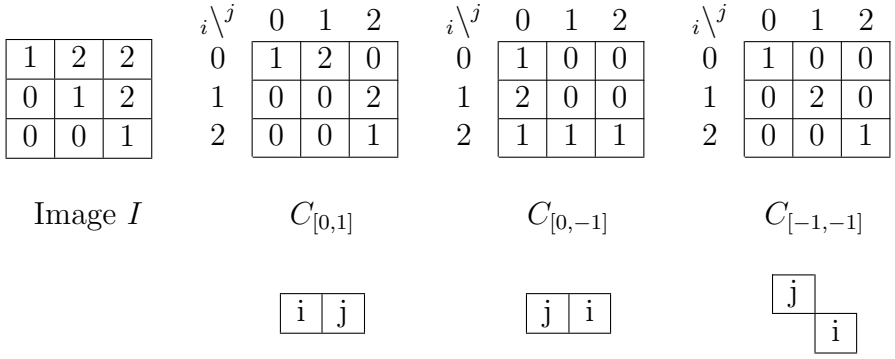


Figure 2.2.1: Toy example of co-occurrence matrices. Shown for an image I of values $\in [0, 1, 2]$. Adapted from [43] L. G. Shapiro and G. C. StockMan, Computer Vision. Upper Saddle River, New Jersey: Prentice Hall, 2001.

$$N_d[i,j] = \frac{C_d[i,j]}{\sum_i \sum_j C_d[i,j]} \tag{2.14}$$

$$S_d[i,j] = C_d[i,j] + C_{-d}[i,j] \tag{2.15}$$

$$Energy = \sum_i \sum_j N_d^2[i,j] \tag{2.16}$$

$$Entropy = -\sum_i \sum_j N_d^2[i,j] \log_2 N_d[i,j] \tag{2.17}$$

$$Contrast = \sum_i \sum_j (i-j)^2 N_d^2[i,j] \tag{2.18}$$

$$Homogeneity = \sum_i \sum_j \frac{N_d^2[i,j]}{1+|i-j|} \tag{2.19}$$

$$Correlation = \frac{\sum_i \sum_j (i-\mu_i)(j-\mu_j) N_d^2[i,j]}{\sigma_i \sigma_j} \tag{2.20}$$

where

$N_d[i,j]$ is the normalized co-occurrence matrix

$S_d[i,j]$ is the symettric co-occurrence matrix

μ_i, σ_i are the mean an standard deviation of the row sum:

$$N_d[i] = \sum_j N_d[i,j]$$

μ_j, σ_j are the mean an standard deviation of the column sum:

$$N_d[j] = \sum_i N_d[i,j]$$

The Laws *Texture Energy*[80], or Laws *Microstructure Method*[46], is a method for extracting texture using 2D filter convolutions. It was created under the observation that the whitened, Laplacian, and Sobel responses of an image "accentuate the microstructure of the texture"[46]. The notation and the tutorial of [43] is used here to explain the process of extracting texture energy. First the source imagery is transformed to zero mean by subtracting the average value in a sliding window. Then the whitened image is convolved with 16 "integer coefficient masks", kernels, created by all combinations of the 4 $[1x5]$ "center-weighted vectors" whose products produce 16 $[5x5]$ kernels, shown in equations 2.21 to 2.24[43]. The individual kernels are named after the two vectors that were used to create it, e.g. L5R5 for the combination of 2.21 and 2.24, the first being vertical and the second horizontal in the

matrix multiplication. The notation of [80] then designates the filter response as $F_{vh}(r, c)$, F_{L5R5} in the previous example. Note that these kernels are separable so that in practice the center-weighted vectors are not necessarily multiplied by each other vector to create the kernel, but for the sake of this discussion the separable quality is omitted. The 16 filter response images are converted to "texture energy image[s]" using equation 2.25[43], of which the bounds result in a 15 x 15 window.

$$L5\ (Level) = [\ \text{-1}\quad \text{-2}\quad 0\quad 2\quad 1\ ] \tag{2.21}$$
$$E5\ (Edge) = [\ \text{-1}\quad \text{-2}\quad 0\quad 2\quad 1\ ] \tag{2.22}$$
$$S5\ (Spot) = [\ \text{-1}\quad 0\quad 2\quad 0\quad \text{-1}\ ] \tag{2.23}$$
$$R5\ (Ripple) = [\ 1\quad \text{-4}\quad 6\quad \text{-4}\quad 1\ ] \tag{2.24}$$

$$E_k[r, c] = \sum_{j=c-7}^{c+7} \sum_{i=r-7}^{r+7} |\ F_k[i, j]\ | \tag{2.25}$$

where

$E_k[r, c]$	is the texture energy map for filter k at pixel $[r, c]$
$F_k[i, j]$	is the response for filter k at pixel $[i, j]$

"An optional next step is to [reduce the feature space by]... normalization, rotational averaging, and extraction of principle components"[80]. The normalization step can be used to reduce the affects of lighting, the rotational averaging can be used to remove redundancies due to filters reused through prior rotation, and the principle component methods seek to find a smaller and more representative feature space on the whole. [43] reduces the 16 texture energy maps E_k are by taking the averages of the symmetric pairs, resulting in the 9 maps:

$L5E5/E5L5$	$L5R5/R5L5$	$E5, E5$
$L5S5/S5L5$	$E5R5/E5R5$	$R5, R5$
$E5S5/S5E5$	$S5R5/R5S5$	$S5, R5$

The symmetric pairs are combined to achieve rotational invariance as one encodes vertical textures, e.g. L5E5, and the other horizontal, e.g. E5L5. The energy maps are produced on a pixel level so that each image forms the familiar image cube of multispectral or hyperspectral sensing. The filter responses can be then appended to form a 1D vector for each pixel. This vector is the textural feature. Laws then applied a classifier to predict the texture from the

vector of filter responses, in his particular case he used a linear discriminant function given a texture mosaic[80].

The creation of the vector of filter responses can similarly be done for a different filter bank of any selection. The *Gabor filter* bank for example uses a "sine-wave grating within an elliptical region", which is swept across a ration and scale resulting in a bank often termed a "daisy petal filter"[46]. The general case including rotation used by OpenCV given in equations 2.26[39]. There are minor variations of this formula, and here one sigma is used for both directions of the Gaussian part, resulting in 5 input variables. Given the high dimensionality of this filter, selecting an appropriate filter set that is not too large as too cause too large a computational load is necessary. Guo for example selected a reduced set of Gabor filters from a more common 24 to a 7[81]. Guo then used mean shift clustering to perform the classification step, also given a mosaic of textures similar to Laws.

$$
\begin{aligned}
g(x, y, \lambda, \theta, \psi, \sigma, \gamma) &= \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \exp\left[i\left(2\pi\frac{x'}{\lambda} + \psi\right)\right] \\
x' &= x\cos\theta + y\sin\theta \\
y' &= -x\sin\theta + y\cos\theta
\end{aligned}
\tag{2.26}
$$

where

λ	is the wavelength of the sinusoid
θ	is the rotation of the filter
ψ	is the phase offset of the sinusoid
σ	is the standard devision of the gaussian envelope
γ	is the spatial aspect ratio of the axis (the elipses aspect ratio)

We can see from the Laws Texture Energy Methods and the Gabor filter set that one method for texture analysis is to use filter response vectors as features to describe texture. However there are many filter banks and variations thereof which often produce a large number of responses. This can be inhibitory when selecting a bank and recognizing the pattern in the responses. Selecting an appropriate filter bank and reducing the dimensionality of the responses is important for any work, but special care is required here as a large number of multispectral bands are included and so it must scale well. This leads us to Shottons Texton implementation[14] and the supporting paper by Jones and Malik[82] regarding such filter bank selection and the classification of the responses.

Textons are used to classify the feature responses into a compact representation, often to be used later on in the process. Textons are the unsupervised clustering of filter responses of source imagery. Textons require the selection of a filter bank, a selection of the bands to apply the filter bank to, and an unsupervised machine learning algorithm to cluster the responses. The clusters then are termed the *texton centers*, or, *texton channels*.

Table 2.2: Original Filter Bank used by Winn, Criminisi, and Minka [12]. Adapted, J. Winn, A. Criminisi, and T. Minka, "Object categorization by learned universal visual dictionary," Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, pp. 1800-1807 Vol. 2, 2005. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1544935.

Function	Scale	Sources Applied To
Gaussian	$\sigma = 1, 2, 4$	L^*, a^*, b^*
First Derivative of Gaussian, X	$\sigma = 2, 4$	L^*
First Derivative of Gaussian, Y	$\sigma = 2, 4$	L^*
Laplacian of Gaussian (LoG)	$\sigma = 1, 2, 4, 8$	L^*

The filter bank used by Shotton[14] was selected using the techniques of Jones and Malik[82] who used filter response vectors to perform stereopsis. The responses were compared using the L_1 norm, where a match signified the pixel correspondence required for disparity production. These filter responses incidentally encode texture. They state that while "one might be tempted to... determine an *optimal* set of filters..." instead "a variety of filter sets would each be adequate, and any good stereo algorithm should not depend critically upon the precise form of the spatial filters chosen"[82](emphasis theirs). A method to evaluate the effectiveness of a filter bank was required, and Singular Value Decomposition (SVD) was used. They posited that a filter set should not include redundant filters, and "even filters for which this is not strictly true, but *almost* true may be a poor choice"[82](emphasis theirs). SVD supplied a quantitative measure of how redundant the filters were. Their formulation used $F^T = U\Sigma V^T$, where $v = F_i^T I$ is the filter response of image patch I for one filter F_i. Each filter F_i was unwrapped into a one dimension vector as was the image patch I. This represents the result for one pixel in the filter convolution/correlation step. Each F_i was then appended to form the matrix F. SVD was applied to the filter matrix which yielded $F^T = U\Sigma V^T$. "The number of non-zero entries in Σ is the rank, r, or the dimension of the vector space spanned by the filters"[82]. They then sought to find a filter bank that was both full rank and the diagonal entries of Σ were $>> 0$. Their toy example illustrating redundancy used rotations of the Gaussian first derivative. The 2D space requires only 2 filters as a linear combination of 2 rotations can sum to any other rotation. Indeed, this is the concept of steerable filters, shown in the x, y transformation of equation 2.26.

The filter bank that was used for the texton implementation of Shotton was selected by Winn, Criminisi, and Minka[12] shown in table 2.2, of which was found to be full rank[14]. The map between the filters in the bank and the source imagery apply them to is also recorded here, of which [12] used the *CIE* $L^*a^*b^*$ space. Shotton used the same filter bank and application map but scaled the σ values to a constant κ. The reasoning for the mapping, where the color channels a^*, b^* where not used except for the Gaussian blurs, was not explained.

Shotton[14] used the Euclidean-distance K-means clustering algorithm to produce the texton centers from the filter responses, reducing the 17 filter responses to a one dimensional

categorical value. Classification of future imagery requires finding the nearest center for the new filter responses. The authors note that a kd-tree algorithm can be used to speed up the classification step during the nearest neighbor search. The authors expanded the texton feature to included context using what they called *Texture-Layout Filters*. The concept behind this filter is that certain textures will be accompanied by other textures near the pixel in question. The texture of a sheep for example can be reasonably expected to be next to grass textures, but not next to jet engines, if that hasn't been observed before that is. The texture-layout feature was the average number of pixels in a nearby rectangle that are a certain texton t. This is shown in equation 2.27[14] for one pixel i and the combination of an area r and texton channel t.

$$v_{[r,t]}(i) = \frac{1}{area(r)} \sum_{j \in (r+i)} [T_j = t] \tag{2.27}$$

where

$v_{[r,t]}$	is the texture-layout filter response
r	is the image region to compute v
t	is the texton center
i	is the pixel that v is computed for
t	is the texton center that the pixel is computed for
T_j	is the texton center at pixel i, $j \in [1, ..., K]$

Variations on the texture layout filters were explored, namely to find the window shape and size with which to search for textons nearby the pixel in question. Integral images for example can be used to efficiently create histograms of the texton centers in an area nearby the center pixel. One further expansion to the texton concept was to learn the layout with respect to the texton at pixel i, such that the response is now $v_{[r,T_i]}(i)$, called texton-dependent layout filters. This formulation was found to be more effective than the texture-layout filter.

2.3 Machine Learning in Practice

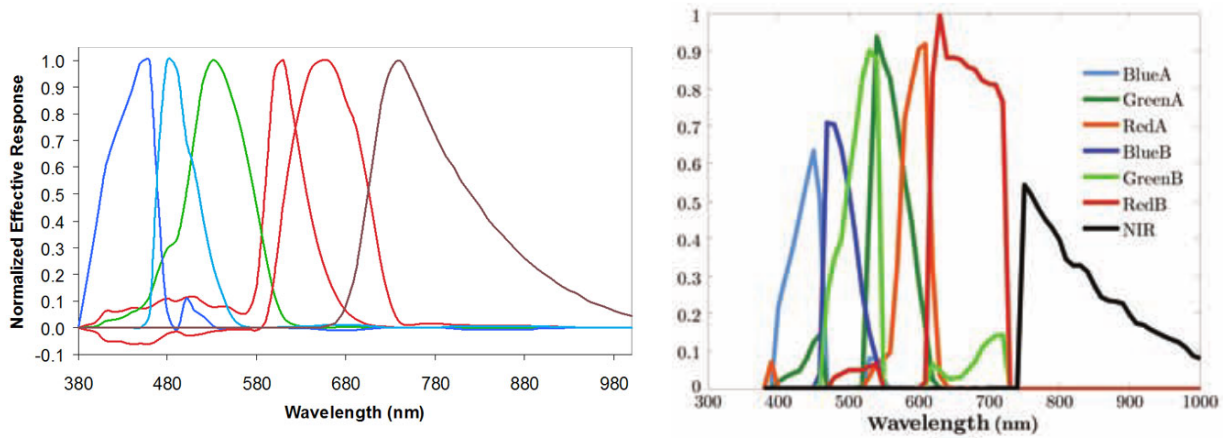
2.3.1 Data Requirements

How much data is required for the desired performance of a machine learning application? It is generally the case that obtaining high quality labeled examples is a time intensive and expensive process, and that a large number of examples are required. There are machine learning methods that seek to reduce the need for labeled examples [34][36], or methods

where high quality synthetic data is substituted for real data[83]. There are also many data sets available on-line that seek to provide labeled data for the benefit of the field such as LabelME[84] and the PASCAL VOC Challenge[85]. On the other end of the spectrum, sifting though a prohibitively large data set to find the most useful instances is in general terms, active search[86][35]. More data can be detrimental as it is costly to maintain and search.

Supervised machine learning requires labeled data to perform. In this section the following questions are addressed: How much data is employed in examples of image labeling in literature? How does one determine if enough data exists for the application?

Namin and Petersson performed pixel level image labeling using a seven band VIS/NIR multispectral camera[1]. Their study is very similar to the goals of this thesis but differ in the bands used. Their system sensed the normal RGB sub-bands and NIR, but also included an additional 3 sub-bands between the pairs B/G, G/R, and R/NIR. The sensitivity to wavelength is illustrated in figure 2.3.1. This study does not have the shifted RGB capability, includes the RGB/NIR band, and adds SWIR and Polar sensitivity.



(a) FD-1665-MS7[87]

(b) Namin and Petersson[1]

Figure 2.3.1: Sensor sensitivity for two 7 band VIS/NIR cameras. For a current camera by FluxData Inc.[87] (left) and for the camera used by Namin and Petersson F. Inc., "The FD-1665-MS7 Seven Channel Camera," 2014. [Online]. Available: http://www.fluxdata.com/products/fd-1665-ms7.
(right)[1] S. T. Namin and L. Petersson, "Classification of materials in natural scenes using multi-spectral images," 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1393-1398, Oct. 2012. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6386074. Used under fair use, 2015.

Namin and Petersson in particular used 10 classes with $15,000$ pixels each for training ($150,000$ pixels total) and $100,000$ random pixels for testing. The use of the number of

pixels as the sample metric here is specific to image labeling: image level and sub-image level object detection will list the sample metric by the number of images or the number of targets in all images, respectively. This makes it difficult to compare results between works. The PASCAL VOC database for example lists data sets for "Classification" on the image level, "Detection" on the sub-image level, and "Segmentation" on the pixel level[88]. The statistics for the segmentation data set are listed as number of images not pixels: 11540 pictures with 27450 instances of targets within those 11540[88]. Finding the number of pixels for testing and training would require parsing the data base. While these figures are beyond the scope of this thesis, a very brief survey is included in table 2.3 in order to provide some insight on pixel level segmentation efforts. The most pertinent is [1] as it documented the set size based on image and pixel number, and this thesis has the most in common with it. Documenting the number of pixels used for this thesis and the number of classes used to train and test each class will be important when comparing this thesis to other works.

Table 2.3: Brief Survey of Data Set Sizes for Pixel Level Segmentation

Data Set	# Images	# Classes	# Pix	Pix/Class
Microsoft Pixel-wise database v1[89]	240	9	-	-
Microsoft Pixel-wise database v2[89]	591	23	-	-
PASCAL VOC 2012 Segmentation[88]	11540	20	-	-
Namin & Petersson[1]	139 Train 30 Test	10	150,000 Train 100,000 Test	15,000 Train

One method of determining if enough data exists centers on the concept of a learning curve. This curve is a measurement of classification performance against the number of training samples[90]. Examples of performance metrics are cross validation accuracy[38], Mean Absolute Error and Root Mean Squared Error[90], F-Score, average precision[91]. Performance metrics are discussed further in 3.4.2 Performance Metrics. The learning curve is typically an exponential rise to an asymptote, where the performance reaches a maximum value as the data set increases and the performance is averaged over the many trials. Formally the learning curve can be modeled by a an inverse power law function shown in 2.28 [90]. The performance Y_{acc} is bounded by [0,1], and is a function of the sample size x and the constants a: minimum achievable error, b: learning rate, and c decay rate[90]. The asymptote is $(1-a)$, or the maximum achievable accuracy. When a classifier converges it means that it can sample the entire training set without making any more updates and the maximum performance has been reached. However when considering if a classifier has converged with respect to the data set size, including additional data does not appreciably improve performance.

$$Y_{acc} = (1 - a) - b * x^c \tag{2.28}$$

Figure 2.3.2 shows an example of a learning curve. The performance rises sharply then reaches the asymptote at 75% accuracy.

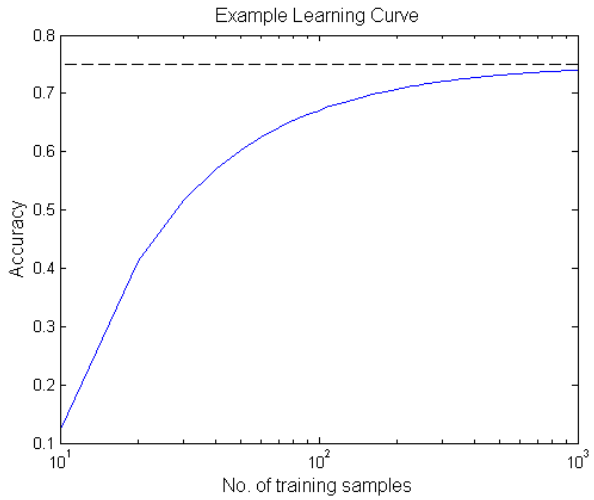


Figure 2.3.2: Toy example learning curve from equation 2.28. $a = 0.25$, $b = 5$, $c = -0.9$.

Figueroa et al. fitted the equation 2.28 to learning curves in machine learning applications in order to predict the performance of a classifier given larger data set sizes[90]. The prediction aspect of this study is beyond the scope of this thesis, but it does illustrate a qualitative metric that can be used. One can estimate the utility of additional data using the learning curve assuming that the performance of classifier converges: additional data will produce diminishing as the number of training samples increases. It will be important to produce the learning curves in order to give insight on the maximum performance and data requirements of a supervised classifier.

2.3.2 The Annotation Process

How are the labels annotated and stored? The method selected here can influence the classes that are capable of being labeled and the burden on the one labeling for accuracy and effort.

Pixels for this work must be labeled on the pixel level given the desired pixel level output. This leaves two main options for encoding the labels: polygons and region masks. LabelMe[84] for example is an on-line database and web tool for annotating images, which stores the annotations as polygons. A list of vertices is stored in XML format along with the object identifier. This method was chosen for portability, one of the main objectives for the LabelMe tool. This method contrasts with the "region masks" as described by [84] that the pixel level Microsoft Research[89] data sets use. Such masks store the labels in terms of the colors, or pixel values. This is similar to the PASCAL VOC challenges for segmentation [45] seen in figure 1.3.7. While these two methods both can be used to produce regions, the polygon

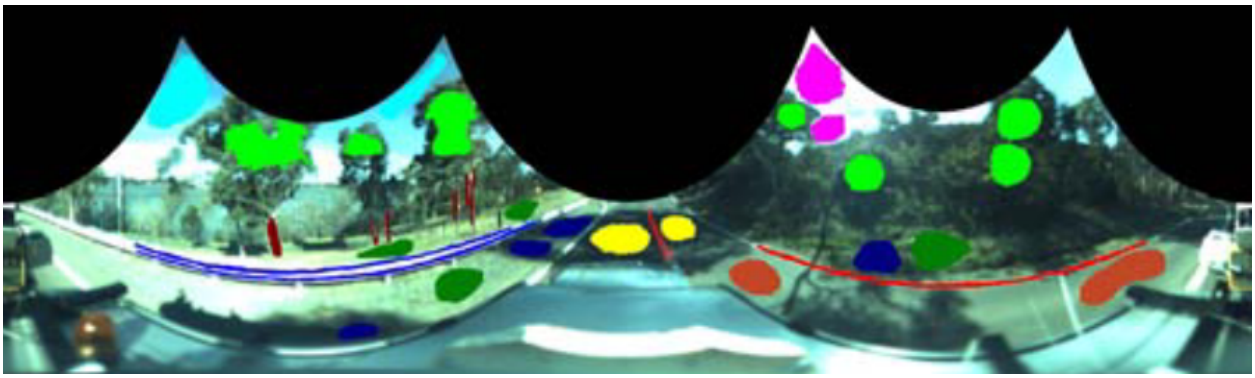


Figure 2.3.3: Example of a hand labeled image for material classification. The labels are seen as the patches of continuous and highly saturated hue. The positive classes are labeled with a generous space between the label and the object boundaries, which reduces the effort and precision required by the user while labeling [1]. S. T. Namin and L. Petersson, "Classification of materials in natural scenes using multi-spectral images," 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1393-1398, Oct. 2012. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6386074. Used under fair use, 2015.

method can describe individual objects which may or may not be occluded. In fact object occlusion/ overlapping polygons was a significant issue with LabelMe of which they addressed by attempts to predict the foreground and background objects. Segmentation masks were then produced using this prediction. The PASCAL VOC[45] segmentation challenge sought to solve this problem by including *object* and *instance* masks. Methods for labeling on the object level are necessary when object detection is required but not for material classification.

How are classes that are not considered labeled? One strategy is to include a *background* class which is effectively an 'else' class, of which [45] used. A void class not used for training or testing can also be included, either explicitly labeled as [45] has done, or implicitly labeled as black as [89] has done. The void class is important here in order to accommodate difficulty in labeling the object edges, of which was the main concern for [45], ween in figure 1.3.7. Namin & Petersson[1] took the concept of the void class one step further and labeled only the positive examples that were easily annotated. Out of the large resolution imagery captured from their hardware only small portions were labeled. The annotations appear as blobs that are well within the object boundaries seen in figure 2.3.3. This strategy reduces the effort required by the labeler in that complex object boundaries are not required.

2.3.3 Class Selection

Namin and Petersson used 10 material classes shown in table 2.4. Care was taken in order to identify objects in shadow with the goal of being robust to lighting differences. Grass

and road were separated into the class within and without shadows. These two classes were "readily marked up" but other classes were not separated as such due to "difficulties in consistent manual labeling"[1].

Table 2.4: Class list used by Namin & Petersson[1]. Adapted, S. T. Namin and L. Petersson, "Classification of materials in natural scenes using multi-spectral images," 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1393-1398, Oct. 2012. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6386074.

Class Index	Class Name	Color
1	Tree Trunks	Brown
2	Light Poles/ Road Guards	Blue
3	Shadow on Grass	Dark Blue
4	Grass	Dark Green
5	Road	Brown
6	White Lines on the Road	Red
7	Shadow on Road	Yellow
8	Leaves	Green
9	Sky	Light Blue
10	Clouds and White Regions in the Sky	Purple

Also note that the sky was separated into clouds or white regions, and else. This is similar to [11] the sky was designated as overcast, clear, or partially cloudy. The latter system can be difficult to identify in the VIS imagery if the exposure was not tuned for the sky. An overexposed sky looks very similar to an overcast or cloudy region.

The vegetation was separated between leaves and grasses for Namin & Peterssons data. Shadow on grass was included but not shadow on leaves due to difficulty in labeling. Indeed, leaves in and out of shadow on tree canopies are often thoroughly mixed over the canopy to the porosity. The regions labeled as tree leaves likely did not include porous regions, i.e. regions were the sky is visible through the leaves, given the single training image and single classified image available in the paper. This is difficult to say given the lack of imagery available and the fact that it was not mentioned explicitly however. Contra to Namin & Petersson are the LabelME annotations which show examples where trees are outlined roughly with the polygon if leaves are present and outlined finely if the tree is bare, i.e. only the branches are segmented. This however is up to the user, and the example mentioned here is from their article on the subject[84].

The main motivation behind the class selection of Namin & Petersson seem to be the inclusion of targets if the labeling process will be straightforward and allow for "consistent manual labeling". They attempted to create a classifier that was robust to lighting, particularly in the texture production step, and attempted to extend that to the class selection by including

the classes separated by in/out of shadow. Selecting classes on this basis and labeling only the portions that were easily and consistently annotated likely reduced the burden on the labeler by a significant amount.

Chapter 3

Theory

3.1 Image Registration

Image registration is the process of spatially warping one image to align with another such that pixels at the same image location represent the same point in space. The bands of light used in this work cannot necessarily be sensed by the same camera due to cost, sensitivity of the sensor, and differences in the way the light interacts with the optics, e.g. LWIR light is blocked by the glass used in optical (RGB) cameras. Therefore multiple cameras must be used, and the multiple viewpoints result in disparity of objects between the viewpoints. The imagery must be processed prior to pixel level fusion such that each point in the registered image represents the same point in space. The registration process begins with the matching of features between the *reference image* and the *sensed image*. The corresponding points are used to determine the parameters in the generic transformation function that will extend the mapping to the entire image. That is, the matched feature points allows one to solve a system of equations for the transformation parameters. While a small number of points are needed to determine a unique solution to the transformation function, e.g. 6 parameters (3 points) for the affine transformation, points on the order of 10^1 to 10^2 are used in the presence of noise. The final transformation function warps the sensed image to align over the reference image.

The *affine* transformation is acceptable when the "scene is very far from the camera"[51]. It is widely used by remote sensing via satellites and aerial vehicles. The *projective* transformation is applicable if the scene is mostly flat and straight lines are preserved between the two images, i.e. the difference in distances from the camera to the objects in the scene are small. The generic projective transformation between two image coordinates (x, y) & (x', y') is shown in equation 3.1 which requires a minimum of 8 parameters to solve[51].

$$x' = \frac{ax + by + c}{dx + ey + 1} \qquad y' = \frac{fx + gy + h}{dx + ey + 1} \tag{3.1}$$

The affine transformation is not sufficient for the scenes likely to be encountered by a UGV as the typical objects are not sufficiently far away. The projective transformation is the next available option in terms of the complexity, and still preserves straight lines. Nonlinear transformations are the next step in terms of complexity but this causes difficulty in the feature correspondence requirement and was not explored. Preliminary results indicated that increasing the complexity of the transform causes the required number of points to significantly increase. Furthermore the points need to be distributed well across the image or the areas that did not have point correspondence would not be registered properly. This can be difficult as there are significant regions of the images that have a non descriptive textures, preventing point correspondence. The projective transformation was chosen in order to reduce the burden on the registration software for producing high quality and numerous matches.

While image registration is an essential tool of this work it is not explored in great depth: the main focus is on the image labeling problem, computer vision, and machine learning. Entire theses could be devoted to registration especially in regards to multispectral data. Traditional registration techniques compare two scenes of the same modality using features that depend on consistent responses, such as direct pixel comparisons or edge and gradient information. Multispectral imagery however often results in wildly different responses, and so intensity information alone is insufficient. Features must be created that are consistent between the modalities. LWIR/VIS registration is a prime example[92][53], where edge features that respond to both modalities similarly are sought after. Therefore, registration code from OpenCV[39] was implemented to produce the registered frames. The overall registration process is shown here in algorithm 2[51], notes on implementation is shown in 4.1, and the results are discussed in 7.2.

Algorithm 2 High Level Description of the Registration Process

1: *Preprocessing*: noise removal, rescaling, etc.
2: *Feature Extraction*: create features for the sensed and reference images.
3: *Feature Correspondence*: match the features between the sensed and reference images.
4: *Transformation Function Solution*: find the transformation parameters by reconciling the matched features with the mathematical model. Filter out noisy matches in the process.
5: *Warping and Resampling*: For every reference pixel, find the pixel location in the sensed image. Extract the warped pixel intensities, sub pixel locations in the sensed images will require re-sampling of the sensed image.

3.2 Features

3.2.1 Texture

Texture is feature that measures the "arrangement of intensities in a region"[43]. Figure 3.2.1 shows three examples of individual textures, in the form of material swatches. Several textures may have the same distribution of intensities in an image region but can be distinct: a black and white checkerboard or striped pattern for example may both have a 50/50 split of black and white but both are distinct textures. Common textures in this study come from foliage, clouds, and drivable surfaces.

(a) Brick (b) Grass (c) Gravel

Figure 3.2.1: Examples of texture. From the University of Southern California Signal and Image Processing Institute, Volume 1: Textures, section 1.4 & 1.5[78]. A. G. Weber, "USC-SIPI Report #315 The USC-SIPI Image Database: Version 5, "Signal and Image Processing Institute University of Southern California Department of Electrical Engineering, Los Angeles, CA, Tech. Rep. October 1997, 2006. [Online]. Available: http://sipi.usc.edu/database/SIPI_Database.pdf. Used under fair use, 2015.

Creating a vector from many filter responses is one way to quantitatively measure texture. While the responses to one filter may not be discriminative several filters at different scales can be viable. Discussion of this topic will require the definition of the following concepts: *image filter*, filter *correlation* and *convolution*, *filter banks*, and *sufficiently representative* filter banks.

Filters and Filter Responses

A filter response is computed from a linear combination between a filter kernel and the pixels neighboring each source pixel. This operation is *cross correlation* or *convolution*. The filter kernel is typically on the order of 10^0 *to* 10^1 pixels wide. A Sobel operator for example produces an estimation of the gradient, shown in equation 3.2[43].

$$S_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \qquad S_y = \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array} \tag{3.2}$$

The filter responses are computed through the *cross correlation* or *convolution* given in equations 3.3, 3.4[43].

Cross Correlation

$$\begin{aligned} G[x,y] &= I[x,y] \otimes F[x,y] \\ &= \sum_{i=-w/2}^{w/2} \sum_{j=-h/2}^{h/2} I[x+i, y+j] F[i,j] \end{aligned} \tag{3.3}$$

Convolution

$$\begin{aligned} G[x,y] &= I[x,y] \star F[x,y] \\ &= \sum_{i=-w/2}^{w/2} \sum_{j=-h/2}^{h/2} I[x-i, y-j] F[i,j] \end{aligned} \tag{3.4}$$

where

F	filter kernel, or mask	w	filter width, odd integer
I	source image	h	filter height, odd integer

The filter response is computed by sliding the filter across the source image, computing the dot product between the filter and the image, then storing result in the same location as the source. The convolution is equal to the cross correlation if the filter is flipped across the horizontal and vertical axis. If the kernel is symmetric then the cross correlation and the convolution will produce the same result. Figure 3.2.2 illustrates why this is the case; additionally note the change of sign between equations 3.3 and 3.4. The cross correlation reverses the response while the convolution does not. The terms cross correlation and convolution are often used interchangeably due to the similarity[43].

$$\begin{array}{|c|c|c|c|c|}\hline 0 & 0 & 0 & 0 & 0 \\ \hline 0 & i & h & g & 0 \\ \hline 0 & f & e & d & 0 \\ \hline 0 & c & b & a & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline \end{array} \quad = \quad \begin{array}{|c|c|c|c|c|}\hline 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline \end{array} \quad \otimes \quad \begin{array}{|c|c|c|}\hline a & b & c \\ \hline d & e & f \\ \hline g & h & i \\ \hline \end{array}$$

$G[x,y]$ $\qquad$ $I[x,y]$ $\qquad$ $F[x,y]$

Figure 3.2.2: Illustration of cross correlation for filter convolutions. The cross correlation is the same as convolution if the kernel $H[x,y]$ is flipped about the horizontal and vertical axis, as the cross correlation reverses the response.

Filter Bank

Multiple filters at different orientations and scales have been used as a feature in stereo matching[82]. While one filter is not distinctive, multiple filters can be appended to a vector producing a discriminative feature list. The filters vary by the mathematical function, said function parameters, the width and height in pixels, and the angle of orientation. Please refer to Filter Implementation 4.2 for specifics in these variations for the filters bank used in this work. Instead addressed here is the question of how to choose the filter bank. Jones and Malik have posited that one should not attempt to determine the optimum set of filters, rather, one should find a set of filters that is sufficiently representative and not redundant: "Any filter that can be expressed as the weighted sum of others in the set is redundant"[82]. Filters can be combined linearly to produce new filters, notably in the *steerable filters* shown in equation 3.5[82][93]. The concept of steerable filters is often used to produce an oriented Derivative of Gaussian filter given the X, Y Derivatives of Gaussian. Jones and Malik note that the because only two orthonormal Derivatives of Gaussians are needed to describe and arbitrarily oriented Derivative of Gaussian, filters beyond the first two will "carry no additional information"[82]. Therefore a filter bank should use two orthogonal first Derivative of Gaussians and no more.

$$F^{\theta} = F^{0^{\circ}} \cos\theta + F^{90^{\circ}} \sin\theta \tag{3.5}$$

where

F^{θ}	2D filter for an arbitrary orientation, θ
$F^{0^{\circ}}$	2D filter oriented along axis 0
$F^{90^{\circ}}$	2D filter oriented along axis 1

Singular value decomposition can determine if the filters are independent or if they can be expressed as a linear combination of the others. The procedure is shown in algorithm 3[82]. Filters that can be expressed as a linearly sum, or almost be expressed so, are poorly chosen, meaning that the singular values should be non-zero and $>> 0$.

3.2.2 Texton

A *texton* is an unsupervised clustering of filter responses. It is the distillation of many filter responses as high level features such as corners cannot be associated with one response alone. A signature produced by multiple filters differing in scale, order, and orientation can be discriminative however. Clustering in the high dimensional feature space of the filter responses produces prototypes which are termed *textons*[79]. Unsupervised clustering is performed on the filter responses of a training set in order to obtain a list of texton centers. Future samples are processed with the same filter bank and the index of the nearest texton to the sample response is assigned as the prediction. The resulting image where the pixel values represent the texton index is termed a *texton map*[14].

Algorithm 3 Filter Bank Evaluation with Singular Value Decomposition (SVD)[82]

1: **for** each filter, index i **do**
2: Sample the filter row by row to arrange the filter values in a column vector F_i by of size $nx1$, where n equals the number of pixels in the filter
3: Horizontally append filter F_i into matrix F
4: **end for**
5: Arrange the source image patch of same size as the filter into a column I by sampling image row by row ▷ $\langle I, F_i \rangle$ is the filter response, equivalent to $F_i^T I$ and $F^T I$ is the vector of filter responses
6: Apply SVD to F^T: $F^T = U\Sigma V^T$
7: Any singular values that are zero or close to zero (the diagonal entries of Σ) indicate the presence of redundant filters

3.2.3 Polarization Features

Polarization of light can be produced by reflections off smooth surfaces or by scattering off relatively small or large particles such as air and clouds. The phase, or the direction of the electric field, can change depending on these phenomena. Table 3.1 shows the notation used by the creator of the polarimetric camera used here[6] as well as the notation of this work. The polarimetric camera consists of 4 Firefly MV VIS/NIR cameras[94], 3 beam-splitters, and 4 polarizing filters. The four images listed in table 3.1 enable polarimetric features to be calculated of which are detailed in this section as *polarization contrast*, *phase*, *Degree of Polarization (DoP)*, *DoLP*, and the *Stokes Vectors*. When referring to the individual cameras the following acronyms are used: Polarization Camera 0: 0° ($PI_{0°}$), Polarization Camera 1: 90° ($PI_{90°}$), Polarization Camera 2: 45° ($PI_{45°}$), Polarization Camera 3: −45° ($PI_{-45°}$).

Table 3.1: Notation for the polarimetric camera, composed of 4 Firefly MV cameras and polarizing filters[6], including the acronym used in this work.

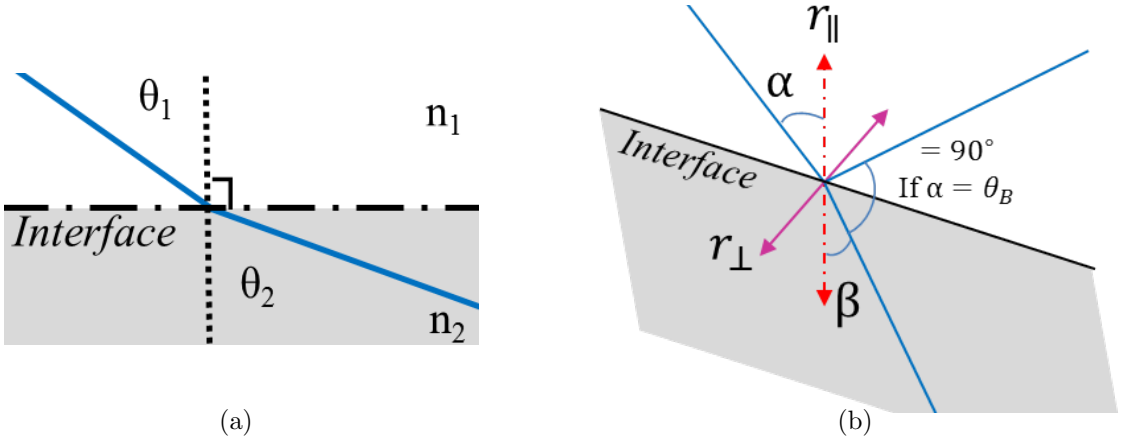
Camera No.	Intensity No.	Polarization Orientation	Acronym
Camera 0	I_0	0° (Horizontal)	$PI_{0°}$
Camera 1	I_1	90° (Vertical)	$PI_{90°}$
Camera 2	I_2	45°	$PI_{45°}$
Camera 3	I_3	−45°	$PI_{-45°}$

Polarization by Reflection

The intensity of unpolarized light reflecting off a surface depends on the angle of incidence and the index of refraction for the two materials. The reflection intensity varies between $0\% - 100\%$ of the incident light, and more light is reflected for angles of polarization that are perpendicular to the plane of incidence, the plane of this page for figure 3.2.3a. This is

shown by equations 3.6,3.8,3.7[29]. The plots in figure 3.2.4 show that the reflected light off of an air/water interface will be completely linearly polarized at the Brewster's Angle and partially polarized else.

This is important because standing water has the potential to exhibit separable features with regards to the DoP, the DoP increasing as the Brewster's Angle is approached. This however is dependent on the angle of the sun and the angle of the camera relative to the water surface. It may be necessary to mount the polarimetric camera "as high as possible to create a viewing angle closer to Brewster's angle"[6] in order to detect this feature, of which the authors' preliminary tests on mud detection may indicate.



(a) (b)

Figure 3.2.3: Supporting graphic for Snell's Law. The faster medium is n_1, i.e. $n_1 < n_2$, and the plane of incidence is the plane of this page in 3.2.3a. Supporting graphic for the reflection coefficients showing the $r_\perp$ and $r_\parallel$ as well as the angle of incidence α, the angle of transmission β in 3.2.3b, and Brewster's Angle θ_B. The vector $r_\perp$ represents the direction of a phase that is perpendicular to the plane of incidence, and $r_\parallel$ for a phase perpendicular to the normal vector of the plane of incidence.

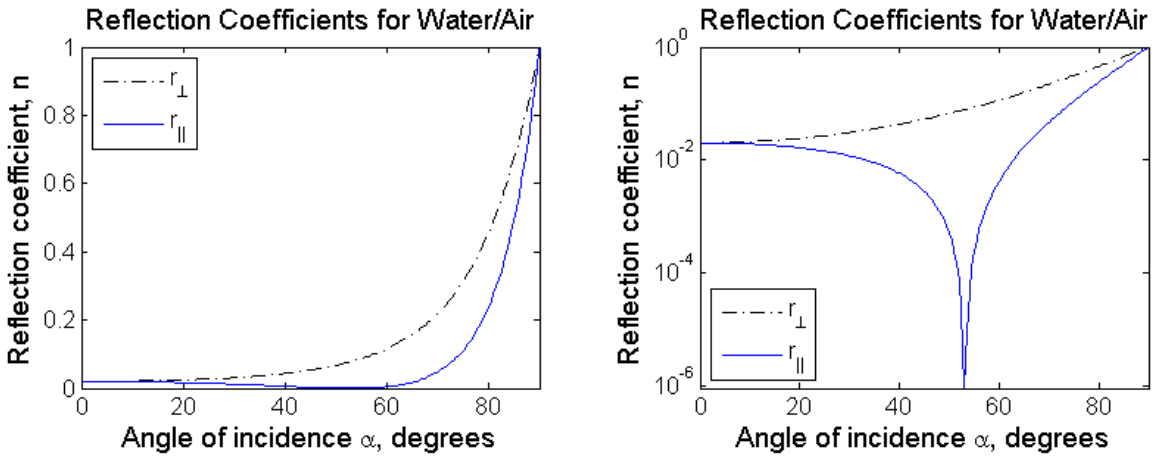
$$\text{Snell's Law} \qquad \frac{n_1}{n_2} = \frac{\sin\theta_1}{\sin\theta_2} \tag{3.6}$$

$$\text{Reflection Coefficient}\perp \qquad r_\perp = \frac{\sin^2(\alpha-\beta)}{\sin^2(\alpha+\beta)} \tag{3.7}$$

$$\text{Reflection Coefficient} \parallel \qquad r_\parallel = \frac{\tan^2(\alpha-\beta)}{\tan^2(\alpha+\beta)} \tag{3.8}$$

where

α	angle of incidence
β	angle of transmission



(a) (b)

Figure 3.2.4: Reflection coefficients for a water/air interface. The intensity of reflected light is shown as a function of angle of incidence in terms of percentage reflected: 1 as 100% reflection. Shown for light perpendicular ($r_\perp$) and light parallel ($r_\parallel$) to the plane of incidence. Linear 3.2.4a, semi-log 3.2.4b. Note that the Brewster's Angle occurs where $r_\perp$ is zero, at 53° for the water/air interface.

Polarization by Scattering

Light is scattered by the air and water particles in the sky. The blue sky is partially linearly polarized, described by *Rayleigh Scattering*[29], and is oriented primarily vertically[5]. Clouds are unpolarized and is described by *Mie Scattering*[29]. Furthermore, skylight is "nearly unpolarized in the vicinity of the sun becoming [increasingly] polarized moving angularly away from the sun"[5]. Reflections on water of the polarized light from the sky and unpolarized light from the clouds result in the light oriented between "$40 - 60°$" for skylight and light oriented horizontally for clouds.

Polarization Contrast

Polarization Contrast is computed by the absolute difference over the sum in intensities of two polarization images in which the polarization filters differ by 90°[4], shown in equation 3.9. The polarization contrast will equal DoLP if one of the images is aligned with the orientation of the component that is linearly polarized, and will be a lower bound else[4][95].

$$P_{contrast}(I_\theta, I_{\theta+90°}) = \frac{|I_\theta - I_{\theta+90°}|}{I_\theta + I_{\theta+90°}} \tag{3.9}$$

Preliminary results for the polarization contrast images are shown in figure 3.2.5, and are discussed at the end of this section. Polarization Contrast images are referred to using the fol-

lowing notation: Polarization Contrast I_0, I_{90} ($PC_{0,90}$), Polarization Contrast I_0, I_{45} ($PC_{0,45}$), Polarization Contrast I_0, I_{-45} ($PC_{0,-45}$).

Phase

The *phase* of polarized light is the direction of the electric field and is perpendicular to the direction of travel as discussed in 1.3.2. It is also known as *polarization direction*[95]. It is calculated by equation 3.10[5]. The "+90°" in equation 3.10 has been included so that the "phase angle is parallel with the linear polarized component"[5]. Otherwise the phase angle would represent the angle where the minimum intensity was observed, perpendicular to the polarized component.

$$\theta_{phase} = \frac{1}{2}\arctan\left(\frac{I_{0^\circ} + I_{90^\circ} - 2I_{45^\circ}}{I_{90^\circ} - I_{0^\circ}}\right) + 90^\circ \tag{3.10}$$
$$if(I_{45^\circ} < I_{0^\circ})\{ \quad if(I_{45^\circ} < I_{0^\circ})\{\ \theta_{phase} += 90^\circ\ \} \quad else\{\ \theta_{phase} -= 90^\circ\} \quad \}$$

DOLP/DOP & Stokes Vectors

The state of polarization can be completely described by the four *Stokes Parameters* which are arranged into the *Stokes Vector* shown in equation 3.11[95]. The Stokes Parameters each have a unit of irradiance: W/m^2[95]. The notation of I_R and I_L denotes circularly polarized light, for right and left hand rotation. As stated in 1.3.2, when the amplitudes of the two fields $\vec{E}$ and $\vec{B}$ are equal and 90° out of phase the light is circularly polarized, and the polarization angle changes with respect to time. I_{0°, I_{45°, I_{90° I_{-45° denote the intensities of the linearly polarized components measured at the angles given in the subscript relative to the 0 component. The $-45°$ is equal to the 135° as the polarization is measured as a plane. S_0 describes the intensity of the scene without polarization information, and the latter three are described in terms of differences between polarization states.

The DoP can be computed with equation 3.13. The DoLP also is a measure of polarization, but does not include the terms that describe the circularly or elliptical polarized light, shown in equation 3.14. The S_3' is not always measured in polarization cameras as circular polarization "does not occur in many places in nature"[6] and measuring DoP alone can be sufficient[5]. This was the case for the design of the polarimetric camera used in this work, which instead of measuring circularly polarized light measures I_{-45°. This was done because "there are very few instances in nature where naturally occurring circular polarization is found...[and] linear polarization is the primary modality of interest [for UGVs]"[6]. The Stokes vector was simplified using the I_{-45° term to produce equation 3.12, which when substituted for the full stokes parameters in 3.13 will result in a measure of linear polarization as the circular & elliptical data is not available[6]. This is formalized in 3.15, which is the measurement of polarization used for this work.

$$\text{Full Stokes} \quad \begin{bmatrix} S_0' \\ S_1' \\ S_2' \\ S_3' \end{bmatrix} = \begin{bmatrix} I_{0^\circ} + I_{90^\circ} \\ I_{0^\circ} - I_{90^\circ} \\ I_{45^\circ} - I_{-45^\circ} \\ I_R - I_L \end{bmatrix} = \begin{bmatrix} I_u \\ I_{0^\circ} - I_{90^\circ} \\ 2I_{45^\circ} - I_u \\ 2I_R - I_u \end{bmatrix} \tag{3.11}$$

$$\text{Simplified} \quad \begin{bmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \end{bmatrix} = \begin{bmatrix} I_0 + I_1 \\ I_0 - I_1 \\ 2I_2 - (I_0 + I_1) \\ 2I_3 - (I_0 + I_1) \end{bmatrix} = \begin{bmatrix} PI_{0^\circ} + PI_{90^\circ} \\ PI_{0^\circ} - PI_{90^\circ} \\ 2PI_{45^\circ} - (PI_{0^\circ} + PI_{90^\circ}) \\ 2PI_{-45^\circ} - (PI_{0^\circ} + PI_{90^\circ}) \end{bmatrix} \tag{3.12}$$

$$\text{Degree of Polarization (DoP)} = \frac{\sqrt{S_1' + S_2' + S_3'}}{S_0'} \tag{3.13}$$

$$\text{Degree of Linear Polarization (DoLP)} = \frac{\sqrt{S_1' + S_2'}}{S_0'} \tag{3.14}$$

$$\text{Simplified Degree of Linear Polarization (DoLP)} = \frac{\sqrt{S_1 + S_2 + S_3}}{S_0} \tag{3.15}$$

Discussion of Preliminary Results

Figure 3.2.5 shows preliminary results for phase, DoLP and the 3 polarization contrast images. The phase image shows that reflections off of anything other than the sky have angles near the horizontal ($0^\circ = 0$ and $180^\circ = 255$). The reflections off the sky are gray meaning that the measured phase is near the vertical ($90^\circ = 128$). The phase of the reflected objects are not necessarily the same as the source object. This indicates that phase may be able to discriminate between reflections and the source object. The DoLP image shows that reflections off the water are indeed polarized, but so are the regions off the vegetation. It is difficult to tell however where the sun is in relation to the cameras, which could impact this measurement greatly. Mostly here the shadowed region is accentuated and the sky is not strongly polarized. This could be due to the pose of the camera, and the presence of clouds in the sky, respectively. The polarization contrast images are somewhat similar to the DoLP image, although Polarization Contrast I_0, I_{90} ($PC_{0,90}$) does not respond as highly to the shadows.

The preliminary data suggests that phase may be influential in discriminating types of reflections and the sky, and DoLP may be influential when detecting shadowed regions. The pose of the cameras may however be a confounding variable.

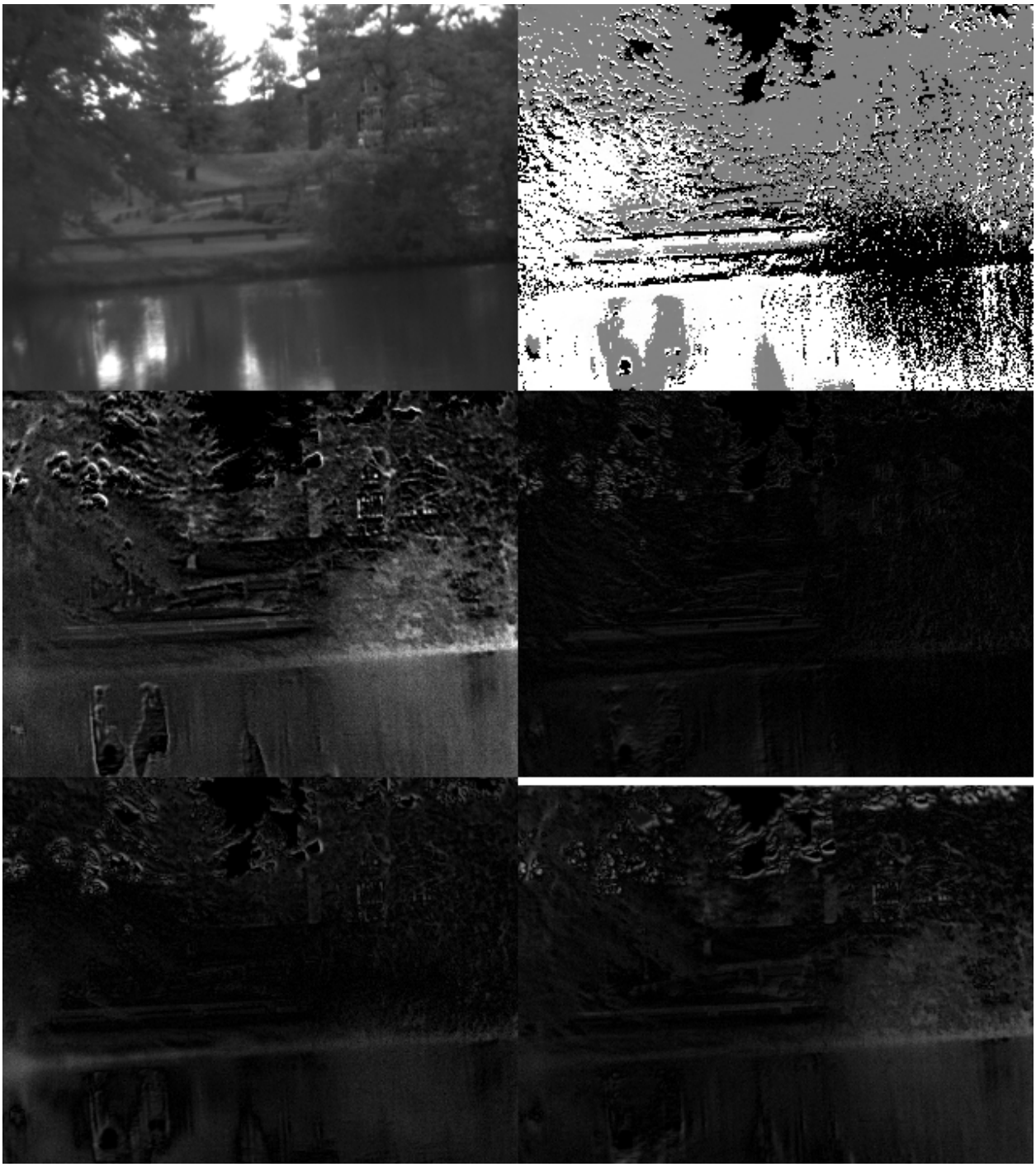


Figure 3.2.5: Preliminary imagery showing polarization contrast. Images are [PI_{0°,*phase*; DoP,$PC_{0,90}$; $PC_{0,45}$,$PC_{0,-45}$]. Using "," as column delimiter and ";" as row. Phase is remapped from $0-180^\circ$ to U8 $0-255$ where 90° is represented by gray. The completely white bar on the top of $PC_{0,-45}$ is outside the field of view and does not represent valid data.

3.3 Classifiers

3.3.1 K-Means

K-Means is an unsupervised machine learning algorithm used to cluster samples based on a distance measurement between said samples. The algorithm outputs the cluster centers $\mu_1, ..., \mu_k$, given the requested number of clusters k and the samples $x_1, ..., x_m$. The algorithm consists of a classification and centering steps shown in pseudo-code 4 with equations 3.16, 3.17. The algorithm optimizes objective $F(\mu, C)$ of equation 3.18 where the optimal K-Means has the smallest distance between the centers and the points assigned to said centers. The square of the l_2-Norm is used here.

Figure 3.3.1 shows a toy example of the initialization, classification, and re-centering step.

Algorithm 4 K-Means

1: **procedure** K-MEANS(*unsigned int* k, *samples* $\vec{X}$)
2: randomly initialize k centers $\mu^{(0)} = \mu_1^{(0)}, ..., \mu_k^{(0)}$ ▷ time step 0
3: **repeat**
4: **Classify**: assign each point $j \in \{1, ..., m\}$ to the closest center eq. 3.17
5: **Re-center**: each $\mu_1, ..., \mu_k$ becomes the centroid of its points eq. 3.16
6: **until** converged;
7: **return** Cluster centers $\mu_1, ..., \mu_k$
8: **end procedure**

$$\mu_i = \frac{1}{k} \sum_{j:C(j)=i} x_j \tag{3.16}$$

$$C^{(t)}(j) \leftarrow \underset{i}{argmin} \left\| \mu_i - x_j \right\|^2 \tag{3.17}$$

$$\min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{j:C(j)=i}^{m} \left\| \mu_i - x_j \right\|^2 \tag{3.18}$$

where

j	index of samples where $j \in \{1, ..., m\}$
i	index of cluster center where $i \in \{1, ..., k\}$
$C^{(t)}(j)$	function of class at iteration t, sample j
μ_i	center at cluster index i
x_j	data at sample index j

(3.19)

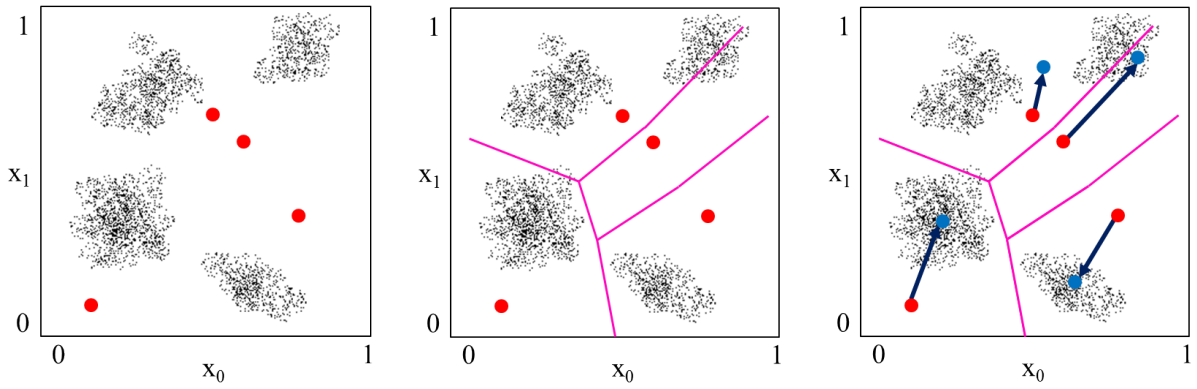


Figure 3.3.1: Toy example of the K-Means algorithm. The cluster centers are initialized randomly 3.3.1a, the samples are assigned to the nearest center 3.3.1b, the centers are recalculated 3.3.1c for each class, repeat 3.3.1b and 3.3.1c until terminated.

3.3.2 Support Vector Machine (SVM)

SVMs are considered by many to be among the best supervised machine learning algorithms that are available *off the shelf* so to speak[96]. SVMs are linear classifiers that separate data by a hyperplane and classify input features based on which side the data point lies about said hyperplane. The optimal SVM seeks to maximize the *margin*, or the distance between the decision boundary and the nearest training point. Although the simplest version of the SVM is a binary linear classifier it has been extended to handle non-separable data, non-linearly-separable data, regression, and multi-class classification. Non-separable and non-linearly-separable data has been enabled through the use of *regularization* and *kernels* respectively[96]. The C-SVM formulation for example allows but penalizes noisy labeling through the regularization parameter C. This section will address the most basic form of the SVM: the *optimal margin classifier*. Concepts related to C-SVM, *kernels*, and multi-class SVM will also be addressed briefly.

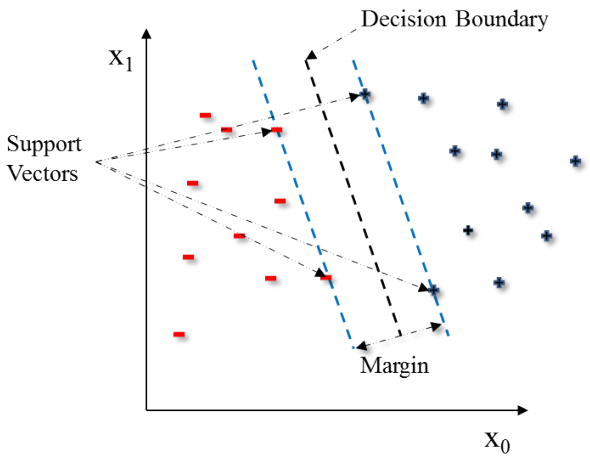


Figure 3.3.2: Generic SVM in two dimensions defining the *decision boundary*, *support vectors*, and the *margin*, for classes $+/-$.

Margins

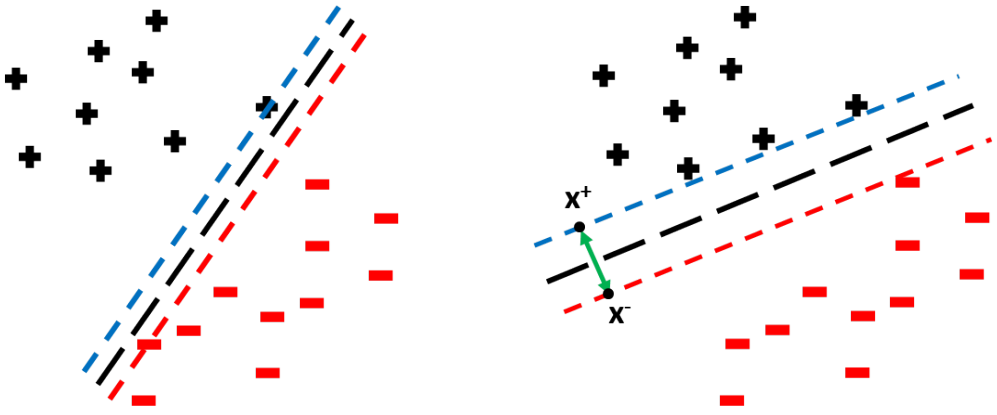
The *optimal margin classifier* first begins with an explanation of the margin. Figure 3.3.2 shows linearly separable data in two dimensions $x_0,\ x_1$ that are separated by the hyperplane of two dimensions, a line. The margin is the distance from the decision boundary to the nearest points which define the support vectors.

The goal is to minimize generalization error, i.e. maximize the future classification accuracy. A small margin is less likely to classify new data given a level of noise as the data is much closer to the hyperplane. Another way to explain this intuition is to consider the distance of the point to the hyperplane as a measure of confidence in the classification[96]: a larger distance equals a larger confidence. Said distance from the hyperplane is obtained by evaluating $(w^T x_i + b)$. This is illustrated in figure 3.3.3.

Base Notation

The classification problem is setup using the notation of equations 3.20-3.22[96]. The binary classification deals with classes of$\{-1, +1\}$, the terms $w,\ b$ denote the notation of a hyperplane with slope vector w and intercept b. Each data instance is one-indexed by parameter i, and there are m features.

Form of data	$\{x_i^{(1)}, ..., x_i^{(m)}\}$	(3.20)
Class Labels	$y_i \in \{-1,\ +1\}$	(3.21)
Generic form of classifier	$\hat{y}_i = h_{x,b}(x_i) = sign(w^T x_i + b)$	(3.22)



(a) (b)

Figure 3.3.3: Illustration of the maximum margin principle for SVM. The data is separated by a decision boundary with a small margin: 3.3.3a, and a large margin: 3.3.3b. The margin width is shown by the arrow between x^+, x^-. A maximum margin is preferred given the goal of minimizing generalization error, i.e. the maximizing the future performance.

Functional/Geometric Margins

A *functional margin* is defined below for every sample, as well as the minimum for the training data.

$$\hat{\gamma}^{(i)} = y^{(i)}(w^T x + b) \tag{3.23}$$

$$\hat{\gamma} = \min_{i=1,\dots,n} \hat{\gamma}^{(i)} \tag{3.24}$$

Note that because the classifier 3.22 is dependent on the *sign* of $w^T x + b$, w and b can be multiplied by a constant and still function as intended[96]. The *geometric margin* $\gamma^{(i)}$ is similar but deals with the margins in the absolute distances of the feature space. The margin, for any training sample x_i is $\gamma^{(i)}$, and the closest point on the hyperplane to the sample x_i is $x^{(i)} - \gamma^{(i)} \cdot w/\|w\|$[96]. This is because $w/\|w\|$ is the unit vector normal to the hyperplane. Since this point is on the hyperplane, it must evaluate to 0 in the original classification function $(w^T x_i + b)$, and solved for $\gamma^{(i)}$ seen in 3.25[96].

$$\begin{gathered} w^T \left(x^{(i)} - \gamma^{(i)} \frac{w}{\|w\|} \right) + b = 0 \\ \left(\frac{w}{\|w\|} \right)^T x^{(i)} + \frac{b}{\|w\|} = \gamma^{(i)} \end{gathered} \tag{3.25}$$

The geometric margin above was defined for the positive side of the margin and can be expanded to handle both sides of the hyperplane, shown in 3.26[96]. The geometric margin

of a data set is the margin with the smallest length 3.27. Combining 3.23 and 3.27 shows that the functional margin $\hat{\gamma}^{(i)}$ equals the geometric margin $\gamma^{(i)}$ if $\|w\| = 1$, shown in equation 3.28[96].

$$\gamma^{(i)} = y^{(i)} \left(\left(\frac{w}{\|w\|} \right)^T x^{(i)} + \frac{b}{\|w\|} \right) \tag{3.26}$$

$$\gamma = \min_{i=1,\ldots,n} \gamma^{(i)} \tag{3.27}$$

$$\gamma^{(i)} = \frac{\hat{\gamma}^{(i)}}{\|w\|} \tag{3.28}$$

Optimal Margin Classifier

The original goal was to minimize generalization error. The derivation so far shows that the *geometric margin* must be maximized. Both the functional and geometric margins can be scaled without changing the decision function 3.22. Therefore the functional margin can become "arbitrarily large without ... changing anything meaningful"[96]. Constraining $\|w\| = 1$ allows one to maximize the geometric margin as well as ensure that distances measured along the hyperplane are not due to some arbitrary scaling. Maximizing the geometric margin gives the optimization problem 3.29[96].

$$\begin{aligned} &\max_{\gamma,w,b} \gamma \quad \text{such that} \\ &y^{(i)}(w^T x + b) \geq \gamma, \quad i = 1, \ldots, n \\ &\|w\| = 1 \end{aligned} \tag{3.29}$$

As Andrew Ng explains 3.29 cannot be used in standard optimization programming packages, so we must modify the formulation and the constraints in order to do so[96]. Substituting γ for $\hat{\gamma}/\|w\|$ using equation 3.28 will remove the problematic constraint $\|w\| = 1$ while still maintaining that the geometric margin is maximized, but $\hat{\gamma}^{(i)}/\|w\|$ is not convex (a requirement for certain optimization problems). If the constraint of $\hat{\gamma} = 1$ is added, the variables w, b are effectively scaled and the problem remains the same. However substitution results in the original maximization problem of $max_{\gamma,w,b}\ \hat{\gamma}^{(i)}/\|w\|$ to $max_{\gamma,w,b}\ 1/\|w\|$. "...noting that maximizing $\hat{\gamma}^{(i)}/\|w\|$ is the same thing as minimizing $\|w\|^2$". Equation 3.30 is the final form that meets the requirements for Quadratic Programming: "convex quadratic objective and only linear constraints"[96].

$$\begin{aligned} &\min_{\gamma,w,b} \frac{1}{2}\|w\|^2 \quad \text{such that} \\ &y^{(i)}(w^T x^{(i)} + b) \geq 1, \quad i = 1, \ldots, n \end{aligned} \tag{3.30}$$

Equation 3.30 is a *primal optimization* problem. Although not shown here, the primal optimization problem can be converted into a *dual optimization* problem by constructing the *Lagrangian* of the primal problem and satisfying the Karush-Kuhn Tucker conditions (KKT). The result is equation 3.31 for classification [96]. The dual formulation allows one two write "the entire algorithm in terms off only inner produces between input feature vectors" which is important for the application of kernels[96].

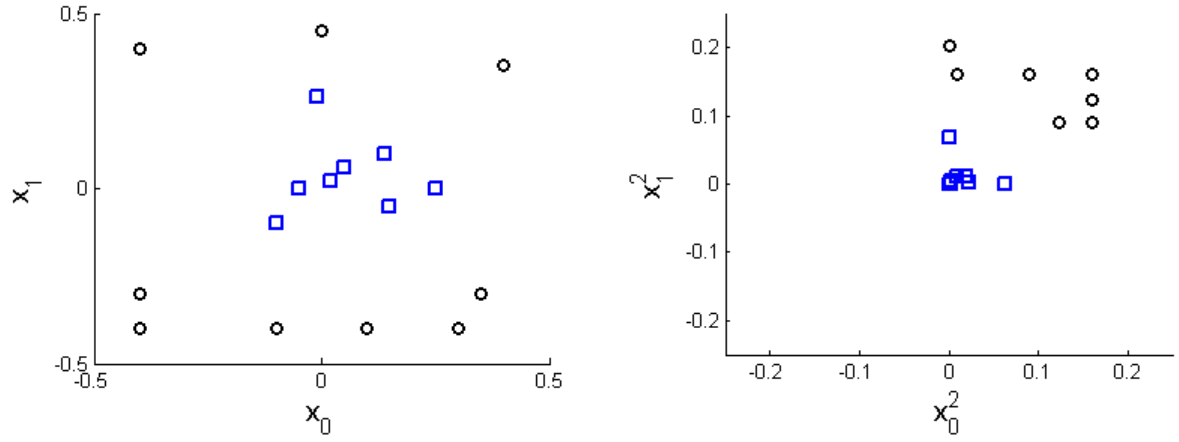
$$\hat{y} = h_{x,b}(x) = sign(w^T x + b) = sign\left(\sum_{i=1}^{n} \alpha_i y^{(i)} \langle x^{(i)}, x \rangle + b\right) \tag{3.31}$$

where

i	index of samples where $i \in \{1, ..., n\}$
α_i	parameter for the dual maximization problem, zero for all training examples except the support vectors.

Kernels

So far discussion of the SVM has been limited to linearly separable data sets. This means that the classes must be able to be separated completely by a hyperplane; $d-1$ dimensions in a d dimension feature space. Kernels are one method to expand the functionality of SVMs to handle non-linearly separable data. Figure 3.3.4 illustrates the concept.



(a) (b)

Figure 3.3.4: Illustration of achieving linear separability with kernels. 3.3.4a shows the data set plotted in (x_0, x_1) space. 3.3.4b plots the same data set in (x_0^2, x_1^2) space. The data is linearly separable in 3.3.4b when non-linearly mapped from 3.3.4a.

Figure 3.3.4a shows two classes of data plotted in (x_0, x_1) space. While the classes cannot be separate by a line, it could be separated by a circle. Plotting instead (x_0^2, x_1^2) yields a plot where the classes can be separated by a line seen in 3.3.4b. Kernels expand the functionality of SVMs by increasing the dimensionality of the feature space to one it is more likely to be linearly separable. The transformation into a higher dimension is termed *feature mapping* and is typically represented by $\phi(x)$[96]. The mapping given in figure 3.3.4 is: $\phi(x)$:

$$\phi(x) = \begin{bmatrix} (x_0)^2 \\ \\ (x_1)^2 \end{bmatrix} \tag{3.32}$$

The features fed to the classifier are then the transformed coordinates ϕ such that $x \rightarrow \phi(x)$. The generic kernel is[13]:

$$K(x, x') = \phi(x)^T \phi(x') \tag{3.33}$$

where

x	the support vector features $x^{(i)}$ in 3.31
x'	the input features for classification x in 3.31

If the feature space or the feature mapping is of a very high dimension, computing $\phi(x)$ can require a significant mount of resources. However, note that 3.31 can be "written entirely in terms of the inner products $[\langle x^{(i)}, x \rangle]$"[96]. This means that any inner product $\langle x^{(i)}, x \rangle$ can be replaced with $K(x, x') = \phi(x)^T \phi(x')$ in 3.31. This is important as the kernel function $K(x, x')$ does not necessarily require one to define $\phi(x)$ explicitly. Simply put, the feature vector is not handled in terms of $\phi(x)$, but rather the inner products $\langle x^{(i)}, x \rangle$ are replaced with a call to the kernel function $K(x, x')$. This is called the *Kernel Trick*[13][96]. Common kernels are shown in equations 3.34, 3.35, 3.36[97], and 3.37[96]. Interestingly, the Gaussian kernel 3.37 corresponds to an "infinite dimensional feature mapping ϕ"[96].

$$K(x, x') = e^{-\gamma \|x - x'\|^2}, \gamma > 0 \qquad \text{RBF} \tag{3.34}$$

$$K(x, x') = (x^T x') \qquad \text{Linear} \tag{3.35}$$

$$K(x, x') = (\gamma x^T x' + r)^d, \gamma > 0 \qquad \text{Polynomial} \tag{3.36}$$

$$K(x, x') = exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right) \qquad \text{Gaussian} \tag{3.37}$$

where

γ, r, d are the respective kernel parameters

Non-separable Data

If the data is not linearly separable even after the use of kernels, meaning that the hard margin is not satisfied $y^{(i)}(w^T x^{(i)} + b) \geq 1$ 3.30, then there will be no solution. The hard constraints $y_i f_i \geq 1$ must be replaced by *soft margin constraints* $y_i f_i \geq 1 - \xi_i$ that penalizes misclassification using *slack variables* $\xi \geq 0$[13] shown by the primal form of C-SVM in equation 3.38[13][38].

$$\begin{aligned} &\underset{w,b,\xi}{max}\ \frac{1}{2}\|w\|^2 + C\sum_i^n \xi_i; \quad \text{such that} \\ &y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i \\ &\xi_i \geq 0, i = 1, ..., n \end{aligned} \tag{3.38}$$

where

$\xi_i \geq 0$	if x^i is on or in the correct margin
$\xi_i = \lvert y_i - f_i \rvert$	otherwise
$C > 0$	is the regularization parameter, a constant

which results in

$0 < \xi_i \leq 1$	when x^i is on the correct side of the boundary but within the margin
$\xi_i > 1$	when x^i is on the wrong side of the boundary

The regularization parameter C controls "the number of errors [that should be tolerated]" where "$\Sigma_i \xi_i$ [is] an upper bound on the number of misclassified points"[13]. C is set using cross validation (section 3.4.1) where large values of C will penalize errors and the margin will be minimized, while small values of C will allow more mis-classifications and maximize the margin.

3.3.3 Binary vs Multi-class

SVMs can extended to multi-class classification using combinations of binary SVMs or methods that modify the objective function to train all classifiers simultaneously.

One vs. rest or *one vs. all* trains an SVM for each class, where every other class is a negative sample for the one being trained. This can however result ambiguous regions where classifiers overlap one another or regions that are not claimed by any classifier[13]. One solution is to choose the most confident classifier by maximizing the distance to the decision boundary: $y^{(i)}(w^T x + b)$ from the optimal margin classifier. The distance from the decision boundary is indeed similar to a measure of confidence given the intuition of maximizing the margin,

however because the objective function $(w^T x + b)$ is linear and can be scaled arbitrarily the distances are not necessarily comparable.

One vs. one or *all pairs* trains a number of classifiers equal to $N(N-1)/2$, the number of pairs between N classes[13] training with the data associated with the pair of classes[38]. The primal problem for a class pair i, j, is defined by 3.39 in the LibSVM package[38], which means that one side of the decision boundary classifies the first class in the pair and the other side belongs to the other class.

$$\begin{aligned} &\underset{w^{ij},b^{ij},\xi^{ij}}{max} \frac{1}{2}(w^{ij})^T w^{ij} + C\sum_{i}^{n}(\xi_i^{ij})_t \quad \text{such that} \\ &(w^{ij})^T\phi(x_t) + b^{ij} \geq 1 - \xi_t^{ij} \text{ if } x_t \in class_i \\ &(w^{ij})^T\phi(x_t) + b^{ij} < -1 + \xi_t^{ij} \text{ if } x_t \in class_j \\ &\xi^{ij} \geq 0 \end{aligned} \tag{3.39}$$

The output classification is the class with the highest number of votes, and ties are solved by choosing "the class appearing first in the array" that stores that class indices[38]. This method can also result in ambiguous regions that are not claimed by any classifier.

3.4 Application and Analysis

3.4.1 Cross Validation

Cross validation can be used to perform model selection. There are often parameters that a learning algorithm requires but one does not know beforehand the values that will produce the best output. K-means for example requires an input of k to define how many classes to compute, and the RBF 3.34 and polynomial 3.36 kernels require an input for γ. One solution would be to perform a parameter search on the entire training set that maximizes classification accuracy of that set. However, the larger goal is to have high classification accuracy for unknown data samples. If model selection is based solely on the data that it was trained on, the resulting model memorizes the training set rather than learning the larger pattern. A measure of performance using the same data would not be indicative of future performance. The error for future data is termed *generalization error*: the "expected value of the misclassification rate when averaged over future data"[13]. The data is therefore broken into two sections: a training set, and a test set. One does not have access to the test set during training by assumption in order to produce a final test accuracy measurement that can generalize well. However, one still needs testing data to perform model selection. Cross validation is performed by further dividing the training set into a smaller sub training set and a *validation* set[13][97]. The models are trained on the cross validation training set

and then evaluated on the validation set. The best models is chosen and retrained on the entire training set. In ν-fold cross validation the training set is divided into ν equal portions. One set is chosen as the validation set to produce the performance metric and the learner is trained on the remaining $\nu - 1$ sets. The performance metric for each cross validation is recorded and combined in some way, such as averaging, to produce a single performance value for the model parameters used. This process is depicted in figure 3.4.1 for 3-fold cross validation. First, the entire data set is divided into the training and testing portions. The training portion is further divided into three folds. Each fold is used as a validation set once, with the remaining 2 folds used to train. *Stratified cross validation* describes folds that have been chosen so that the proportion of each class is roughly equal between folds[13].

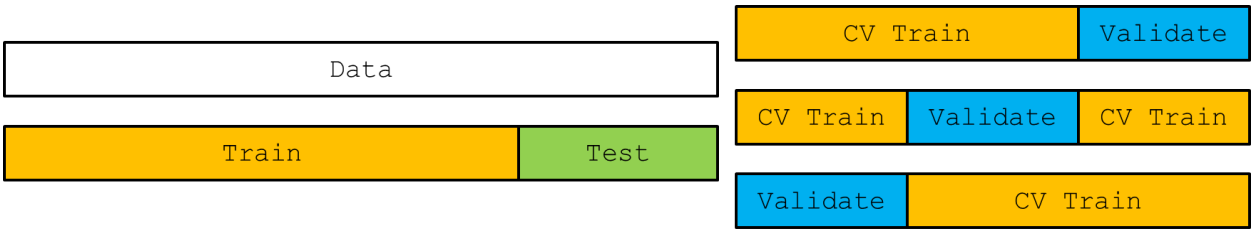


Figure 3.4.1: ν-fold cross validation with three folds. The data (left) is first divided into the training and test set. The training set is divided evenly into 3 folds, using each fold once as the validation set and the remaining $\nu - 1$ folds as the cross validation training set.

3.4.2 Performance Metrics

As described in 3.4.1 the data is divided into a training set and test set and the goal is to optimize a measure of performance on the test set given the training set. The test set is by assumption not available to the user for training[13] because the overall goal is to produce a classifier that generalizes well to future data. Maximizing accuracy on the training set would model the noise rather than the general trend. This is known as the over-fitting problem: minor variations in inputs are more likely due to noise rather than the system at large[13], so a model that has over fitted the training data will not generalize well to new data. Also described in 3.4.1 is the use of a validation set to measure performance before a final model is selected. While the importance of separating data for testing and training has been established, the metrics for performance has so far been omitted. This section addresses common performance metrics, ending with an emphasis on multi-class problems.

Confusion Matrix and its Derivatives

Classification Accuracy is a straightforward metric for performance as it is simply the number of correct predictions divided by the number of samples returned as a percentage: equation 3.40. This is illustrated in table 3.2 Confusion Matrix for Binary Classification adapted from

Table 3.2: Confusion Matrix for Binary Classification, adapted from [13]. K. P. Murphy, Machine Learning A Probabilistic Perspective. Cambridge, Massachusetts: The MIT Press, 2012.

		Prediction 1	Prediction 0	
Truth	1	TP	FN	N_+
	0	FP	TN	N_-
		$\hat{N}_+$	$\hat{N}_-$	

where

TP True Positive — FP False Positive

FN False Negative — TN True Negative

N_+ # true samples — N_- # false samples

$\hat{N}_+$ # predicted true — $\hat{N}_-$ # predicted false

[13], where the individual cell entries are designated as True/False Positive and True/False Negative.

$$Classification\ Accuracy = \frac{\#\ correct\ predictions}{total\ \#\ samples} \tag{3.40}$$

The marginals of the *confusion matrix* show the number of instances in the data set for each class delineating both prediction and ground truth. N_+ in this notation designates the number of samples that are of the positive class and N_- for the negative class. The predicted values are likewise and are designated with the hat ($\hat{}$) operator. The values shown in 3.2 are used in quantifying the *true positive rate*, *false positive rate*, *precision*, and *specificity*[13][98], which are defined in equations 3.41. Equations 3.41a - 3.41d are produced through row normalization of the *confusion matrix* in this case as the ground truth is encoded in the rows.

$$True\ Positive\ Rate = TP/N_+ \tag{3.41a}$$

$$False\ Positive\ Rate = FP/N_- \tag{3.41b}$$

$$Precision = TP/\hat{N}_+ \tag{3.41c}$$

$$Specificity = TN/\hat{N}_- \tag{3.41d}$$

$$Classification\ Accuracy = \frac{TP + TN}{N_+ + N_-} \tag{3.41e}$$

The *true positive rate* is also known as *sensitivity*, *recall*, and *hit rate*[13][98]. *Precision* is the number of positive samples over the number predicted as positive, while the *specificity* is the number of negative samples over the number predicted as negative. These parameters and others can be combined to form a measure of performance tailored to the requirements of the user. A common example of this is the detection of cancer in a patient. It is very costly if the cancer exists and is not detected (a false negative) but it is less costly if the cancer does not exist and is detected (a false positive). The user in this case should favor a high *Specificity* rather than a high *Precision*. The trade-off between false positives and false

negatives is addressed here in relation to Receiver Operating Characteristic (ROC) curves.

False Positive vs. False Negative

Given a binary classification problem, one can assign weights to the false negative and false positive cases in order to represent the cost of that error type. Let L_{FN} be the loss for a false negative and L_{FP} be the loss for a false positive. Equations 3.42 and 3.43 represent the expected loss for the two error types if the classifier outputs the probability of the prediction given the data[13].

$$p(\hat{y} = 0|) = L_{FN}\, p(y = 1|x) \tag{3.42}$$
$$p(\hat{y} = 1|) = L_{FP}\, p(y = 0|x) \tag{3.43}$$

Therefore the class $y = 1$ should be chosen "iff $p(y = 1|x)/p(y = 0|x) > \tau$, where $\tau = c(1 + c)$ [and $L_{FN} = cL_{FP}$]"[13]. Varying the decision threshold τ yields a trade-off between the false positive rate and false negative rate. If these two rates are plotted vs. one another, the ROC is formed. A toy example of this graph is shown in figure 3.4.2.

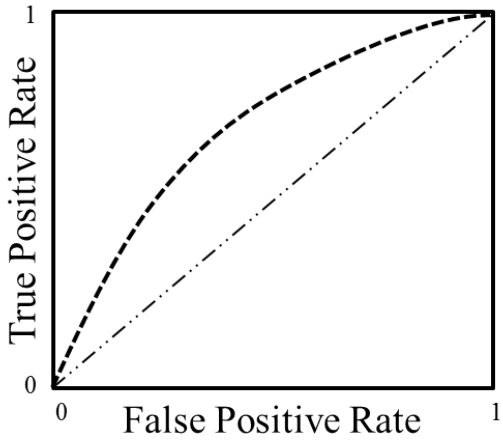


Figure 3.4.2: A toy example of an ROC showing false positive rate vs. true positive rate. The random choice is shown as the diagonal, and the function of the rates as the threshold τ is swept is the dashed line.

The lower left corner is where no positive predictions are issued or gained, the top right is where all points are predicted as positive regardless of the data. The top left corner is the perfect classifier where all points are correctly classified as true with no error. A performance given the decision threshold τ will be higher the closer it is to the point $(0, 1)$, the top left

corner. The ROC curve can be used to visually represent the FP/FN trade-off that results from sweeping τ.

The ROC curve can be distilled into one performance metric called the Area Under the Curve (AUC), where the maximum is 1 and the value should be maximized[13]. The AUC is "equivilant to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance"[98]. While the minimum is technically 0, random choices (for balanced data sets) will be 0.5 which represents the realistic minimum for a classifier.

Applying the ROC and AUC to multi-class problems is difficult as the dimensionality quickly becomes unmanageable. With n classes the confusion matrix is nxn, and there are $n^2 - n$ off diagonal entries (which represents the possible errors) meaning that the ROC space will be $n^2 - n$[98]. For a three class problem the ROC space will have 6 dimensions. This form of analysis is outside the scope of this work as there are simpler metrics, and the need for a preference between false positives or false negatives is not inherent to the general image labeling problem and will require a more specific application to define.

F-Scores

The *F Score* is the harmonic mean of precision and recall, and is used to quantify the performance of one decision threshold τ and the resulting precision and recall values[13]. The maximum value is 1 and the minimum 0. Equation 3.44 is the generic equation that allows for a trade-off between precision and recall[99]. The *F Score* is favored towards precision when $\beta > 1$ and recall when $\beta < 1$. The *F1 Score* weighs precision and recall equally with $\beta = 1$ and is redefined in 3.45 using the notation of the confusion matrix in table 3.2. The harmonic mean is used rather than the arithmetic $(P + R)/2$ in order to produce an more accurate measure if the classifier maximizes recall by simply predicting all instances as positive. In this case the *recall*, or true positive rate, will be 1 but the precision will be ≈ 0, and the arithmetic average will be ≈ 0.5. The harmonic mean will be ≈ 0 instead.

$$F_\beta = \frac{(\beta^2 + 1) * precision * recall}{\beta^2 * precision + recall} \tag{3.44}$$

$$F_1 = \frac{2 * precision * recall}{precision + recall} = \frac{2TP}{2TP + FN + FP} \tag{3.45}$$

An *F Score* will need to be combined in some way in order to handle multi-class classifiers. The options available are *Macro-Average*, *Micro-Average*, *Weighted-Average*, shown in equations 3.46[13], 3.47[13], 3.48[100].

$$Macro - Average \qquad F_{1M} = \frac{1}{\eta}\sum_{c=0}^{\eta-1} F_1(c) \tag{3.46}$$

$$Micro - Average \qquad F_{1m} = \frac{TP_m}{2TP_m + FN_m + FP_m} \tag{3.47}$$

$$Weighted - Average \qquad F_{1W} = \frac{1}{\sum_{c=0}^{\eta-1} N_+(c)}\sum_{c=0}^{\eta-1} F_1(c) * N_+(c) \tag{3.48}$$

where

$$\eta = \#\text{ of classes} \qquad c \in [0, 1, \ldots, \eta - 1] \qquad N_+(c) = \#\text{ instances of } c$$

$$FP_m = \sum_{c=0}^{\eta-1} FP(c) \qquad TP_m = \sum_{c}^{\eta-1} TP(c) \qquad FN_m = \sum_{c=0}^{\eta-1} FN(c)$$

$$F_1(c) = \frac{2TP(c)}{2TP(c) + FN(c) + FP(c)}$$

The *Macro-Average* is simply the average of the F1-Scores. The *Micro-Average* sums up the TP, FN, FP parameters from the confusion matrix for each class and uses those sums in equation 3.45. The *Weighted-Average* weighs each F1 Score by the percentage of the samples that belong to the respective class. The Macro-Average gives equal weight to each class, while the Weighted Average can be used to weigh each class equally given imbalanced data, i.e. unequal values of $N_+(c)$ for each class.

Learning Curves & Convergence

Supervised machine learning requires labeled samples of which can be in shortage as they are often relatively expensive to create. The knowledge of how much annotated data a machine learning solution will require could be useful when determining the feasibility of an application. Unfortunately, it can be common to start with an ad hoc initial data set and gradually increase the samples until a desired performance is reached, "based on a vague but generally correct belief that performance will improve with a larger sample size"[90]. Figueroa, Zeng-Treitler, Kandua, and Ngo explored extrapolations of the learning curve in order to produce the "prediction of a classification performance [given] a sample size"[90]. While such extrapolations are outside the scope of this work the *learning curve* can be an informative graph when trying to determine if a particular learner has converged.

As explained briefly in 2.3.1 a *learning curve* is a plot of the classification performance vs. the training sample size. The typical curve is exponential and asymptotically approaches the

maximum performance $(1 - a)$ as the sample size increases. The generic curve is given by equation 2.28[90] and figure 2.3.2. A learning curve can be used as a qualitative measure of *convergence* in lieu of fitting an inverse power law function 2.28 to the performance as a function of the number of training samples.

3.4.3 Feature Standardization

Feature normalization can be an important preprocessing step for Machine Learning depending on the algorithm used and the nature of the data. It can be referred to as *scaling*, *standardization*, or *normalization*, depending on the field[101]. All refer to some manner of adding/subtracting by a constant and multiplying/dividing by a constant to displace the center and scale the bounds. It is common to rescale the feature space so that the previously measured data is bounded by $[0, 1]$ or $[-1, 1]$, or so that the data is mean zero and standard deviation one[101][37][97]. Rescaling to zero mean and unit standard deviation is referred in this work as standardization. This is performed for each feature dimension separately.

The mean and standard deviation of dimension j is defined as μ^j and σ^j in equations 3.49 and 3.50 respectively. Each dimension of the data set (the column vectors) can be transformed to zero mean and unit standard deviation using equation 3.51. This is referred to as *whitening* and *centering*[31].

It is important that all data fed to the learner is scaled by consistent factors per each dimension. The scaling factors are produced from the training data and are applied to future data prior to prediction. Otherwise scaling the training and test data separately produces an erroneous set[97].

For a set of data $X = [x_0, ..., x_{N-1}]$ of size NxD with N samples and D features, the element x_i^j denotes the value at sample i and in dimension j. The mean μ^j is given by

$$\mu^j = \frac{1}{N} \sum_{i=0}^{N-1} x_i^j \tag{3.49}$$

and the standard deviation σ^j is given by

$$\sigma^j = \sqrt{\frac{1}{N} \sum_{i=0}^{N-1} (x_i^j - \mu^j)^2} \tag{3.50}$$

and the function for normalizing element x_i^j into a zero mean unit standard deviation is

$$z_i^j = \frac{x_i^j - \mu^j}{\sigma^j} \tag{3.51}$$

and is extended to vector notation by

$$\begin{aligned} \vec{x_i} &= [x_i^0, ..., x_i^{D-1}] \\ \vec{\mu} &= [\mu^0, ..., \mu^{D-1}] \\ \vec{\sigma} &= [\sigma^0, ..., \sigma^{D-1}] \\ \vec{z_i} &= (\vec{x_i} - \vec{\mu})./\vec{\sigma} \end{aligned} \tag{3.52}$$

where

operator ./ denotes element-wise division

Feature scaling is necessary when features are combined by a distance function[101]: the RBF kernel for example of equation 3.34 and the K-Means classification function (equation 3.17). Features with relatively small variances will become negligible if compared to features with larger variances. The end result is that features with large variances will be weighted more heavily than the others. Standardizing the features will weigh each variable equally. Feature scaling is not considered necessary if the features are combined linearly[101] as the resulting classifier can nullify the scaling if the classifier converges to appropriate slope and intercept values.

There are many optima functions and the means for finding the local/global optimum are typically numeric[31]. The *steepest descent* algorithm, or *gradient descent* is one of the "simplest algorithms[s] for unconstrained optimization"[31] and is sensitive to scaling [101]. This is because the *gradient descent* algorithm relies on the first derivative to iteratively step along the function until the maximum/minimum is found. Feature standardization in this type of application can decrease computation time.

Chapter 4

Implementation

4.1 Registration Implementation

The images where spatially registered using code from OpenCV[39], and temporally registered through a Pulse Width Modulation (PWM) signal tied to all the trigger lines of the cameras. A brief description of the code is given in algorithm 5 and 6. The images were registered using the OpenCV implementation for Speeded-Up Robust Features (SURF) features and descriptors, a Fast Approximate Nearest Neighbor Search Library (FLANN) descriptor matching algorithm, and a projective transformation. Specifically: cv::SurfFeatureDetector, cv::FlannBasedMatcher, cv::findHomography. The SURF was used for ease of implementation only, as there are many other key point and descriptor algorithms. Oriented Binary Robust Independent Elementary Features (ORB) for example is more recent and is free to use[102]. The polarimetric images, PI_{90°, PI_{45°, PI_{-45° were first registered to PI_{0°. PI_{0° was registered to the VIS/NIR, and the resulting homography matrix was used to register the previously warped PI_{90°, PI_{45°, PI_{-45° to the VIS/NIR. The remaining SWIR was registered to the VIS/NIR frame. The registration was then complete as the VIS/NIR images are registered at acquisition through the use of a beam splitter. Binary masks were produced for each image using the homography matrices to designate the valid FoV, 0 as invalid, 1 as valid. The masks were then combined downstream using an element-wise AND operator to find the smallest valid FoV given a set of cameras.

It is important to note that the registration code used from OpenCV was created to match images typically taken from the same camera. The quintessential challenges of image registration is to be robust to lighting and pose, such as scale and rotation and the lighting effects that result from changing said pose or the camera parameters. Image registration systems often encode edge orientations into descriptors and then computes the distances between descriptor vectors to find a match to another source image. SURF for example encodes "wavelet responses" a.k.a texture into the descriptor, of which the "wavelet responses are invariant

to a bias in illumination (offset)"[103]. Intensities and texture between multispectral bands however are often differ drastically as one can see in chapter 2, notably for texture only there is much more texture in the NIR band than VIS and much less texture in LWIR than any other band. The intensities are often polar opposites as well. As a result, the LWIR band was sufficiently different to prevent its registration in this system, so the band was not used. The SWIR band ($0.9\mu m$ - $1.7\mu m$)[23]) was sufficiently close enough to the NIR band ($0.65\mu m$, $0.76\mu m$ - $1.1\mu m$)[22]) that registration using the method in algorithm 5 was successful. This was sufficient as the NIR band was already registered to the VIS band due to the use of beam splitters in the JAI camera[22]. The registration between the Polarimetric cameras and the NIR camera was straightforward and posed little issue as it was a conglomeration of the VIS and NIR band: $0.42\mu m$ - $1.0\mu m$)[24].

Algorithm 5 Spatial registration between two images

procedure REGISTERIMAGEPAIR($fixed$, $sensed$)
  for each image **do**
    Extract key points
    Extract descriptors
  end for
  $matches = matchDescriptors(fixed.descriptors, sensed.descriptors)$
    ▷ Filter matches based on distance between points
  Find $minimum$, $maximum$ distance between matched points
  $matchMaxThreshold = c_0 * minimum + c_1$ ▷ experimental constants: c_0, c_1
  for all $matches$ **do**
    if $match.distance > matchMaxThreshold$ **then** Discard match
    end if
  end for
  $H = findHomography(matches)$
  $sensedImageWarped = warp(sensedImage, H)$
return $sensedImageWarped$
end procedure

4.2 Filter Implementation

The filter bank was chosen to be the same as Shotton, Winn, Rother, and Criminisi[14] and Winn, Criminisi, and Minka[12] as their bank was determined to have full rank via SVD, using the methods of Jones and Malik[82]. Again, note that "the choice of a filter-bank is somewhat arbitrary as long as it is sufficiently representative"[14]. The filter bank is shown in table 4.1. Winn et al. used a filter bank scale factor $\kappa = 1$ while Shotton et al. varied the value between experiments using $\kappa \in \{1, 0.7, 0.45\}$.

Algorithm 6 Spatial registration for all cameras

procedure REGISTERALL($imageVIS$, $imageNIR$, $imageSWIR$, $imagePolar$)
  ▷ Register polarimetric cameras to PI_{0°
  $subP1 = registerImagePair(imagePolar.P0,\ imagePolar.P1)$
  $subP2 = registerImagePair(imagePolar.P0,\ imagePolar.P2)$
  $subP3 = registerImagePair(imagePolar.P0,\ imagePolar.P3)$
  ▷ Register all cameras to VIS/NIR
  $regP0,\ H_{P_0 \to VIS}$ = registerImagePair($imageVIS$, $imageP0$)
  ▷ Homography P0 to VIS
  $regP1 = warp(subP1,\ H_{P_0 \to VIS})$
  $regP2 = warp(subP2,\ H_{P_0 \to VIS})$
  $regP3 = warp(subP3,\ H_{P_0 \to VIS})$
  $regSWIR$ = registerImagePair($imageVIS$, $imagePolar$)
  for all registered image **do**
    create binary mask of the valid field of view
  end for
return registered images and masks
end procedure

The filters are shown in figure 4.2.1, expanded in size for readability and normalized to fit in an 8-bit gray scale image. The cross sections of the functions used are shown in figure 4.2.2 and in equations 4.1, 4.4[43], 4.2. It is important to note that the filters, or kernels, shown in figure 4.2.1 are normalized such that the range is scaled to the bounds of $[0, 255]$ for human consumption in gray scale format. The actual kernels used however, are normalized such that the discrete integral of the kernel is 1. This should be done when the "actual values output by the filters are important"[104], such as when one blurs an image intended for human consumption. This is accomplished by dividing the kernel by a scalar equal to the summation of the kernel matrix. The filtered images are then easier to inspect, while it is not necessarily required for a learner to function.

Gaussian
$$F_G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{4.1}$$

First Derivative of Gaussian, X
$$\frac{\delta F_G(x, y, \sigma)}{\delta x} = \frac{-x}{2\pi\sigma^4} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{4.2}$$

First Derivative of Gaussian, Y
$$\frac{\delta F_G(x, y, \sigma)}{\delta y} = \frac{-y}{2\pi\sigma^4} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{4.3}$$

Laplacian of Gaussian (LoG)
$$F_{LoG}(x, y, \sigma) = \frac{1}{\pi\sigma^4}\left(1 - \frac{x^2 + y^2}{2\sigma^2}\right) \exp\left(\frac{x^2 + y^2}{2\sigma^2}\right) \tag{4.4}$$

Table 4.1: Original Filter Bank used for Textonization by Shotton, Winn, Rother, and Criminisi[14]. Adapted, J. Shotton, J. Winn, C. Rother, and A. Criminisi, "TextonBoost for Image Understanding : Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture , Layout , and Context," Int. Journal of Computer Vision (IJCV), 2007. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=117885.

Function	Scale	Sources Applied To
Gaussian	$\sigma = \kappa, 2\kappa, 4\kappa$	L^*, a^*, b^*
First Derivative of Gaussian, X	$\sigma = 2\kappa, 4\kappa$	L^*
First Derivative of Gaussian, Y	$\sigma = 2\kappa, 4\kappa$	L^*
Laplacian of Gaussian (LoG)	$\sigma = \kappa, 2\kappa, 4\kappa, 8\kappa$	L^*

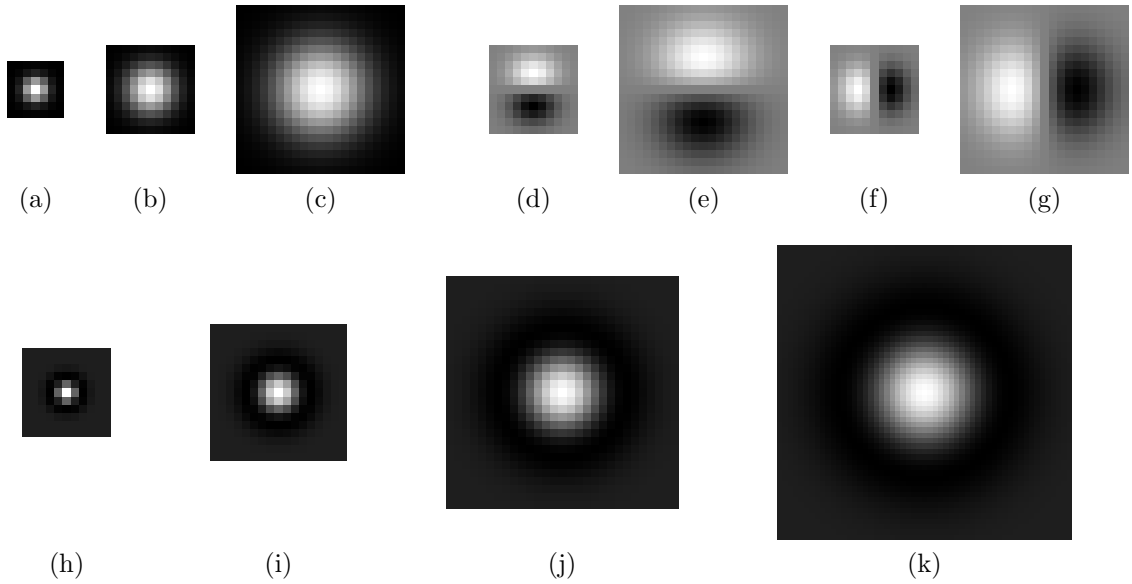


Figure 4.2.1: Filter bank of table 4.1, $\kappa = 1$, increased in size by a factor of 3. Gaussian: 4.2.1a-4.2.1c; First Derivative of Gaussian X: 4.2.1d, 4.2.1e; Y: 4.2.1f, 4.2.1g; LoG: 4.2.1h - 4.2.1k.

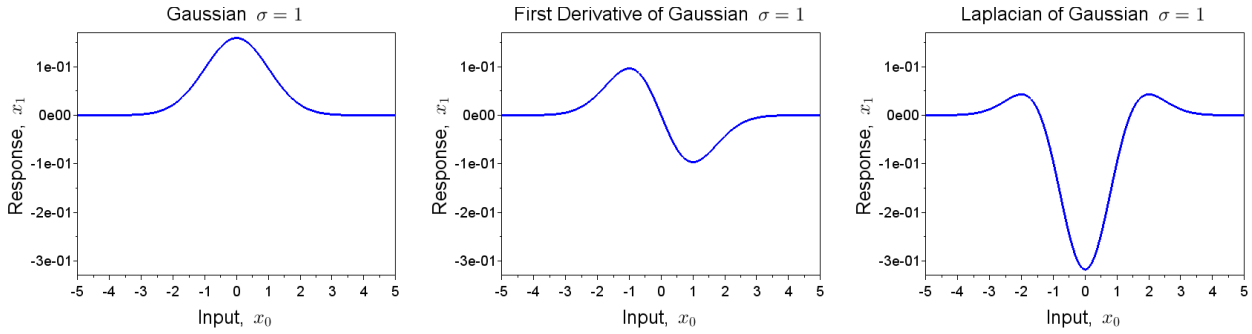


Figure 4.2.2: 1D plots of the functions used for the filter bank of table 4.1.

The Gaussian function blurs the image, with an increasing affect as σ increases. The first derivative of the Gaussian function is a measure of edges. Combining the X and Y derivatives will give a measure of an oriented edge as the two are orthogonal, such as with the steerable filters in 3.5. The Laplacian of Gaussian (LoG) is the second derivative of the Gaussian function, and is also a measure of edges. The LoG produces a high response for "small blobs coinciding with the center lobe, and large step edges [near the center]... [and] zero response on both constant regions and ramps"[43].

The widths of the kernels are sufficiently small with the exception of the LoG, which in this example is 55 pixels wide using $\sigma = 8pixels$ (the smallest, Gaussian $\sigma = 1pixel$ is 7 pixels wide). The width was calculated based on where the Gaussian function equaled an arbitrary error value, in this case 0.01, for the Gaussian and First Derivative of Gaussian. The width of the LoG was calculated to be where the function equaled a percentage of the outer lobe maximum approaching from $+/-\infty$. The percentage cutoff was 0.20 in this case. The large width of the LoG is due to the fact that this function trails off more slowly than the other functions as the inputs approach $+/-\infty$. It appears by visual inspection that there is a perimeter of zero values around the kernels, indicating that they are sufficiently wide as the function is represented. Larger widths negatively affect computation time but it was not necessary to optimize during these experiments. Those interested in minimizing computation time can explore experiments minimizing the kernel widths as well has separable filters. The use of separable filters reduces the 2D convolution into two 1D convolutions.

4.3 Texton Implementation

The texton subsystem was implemented using the filter bank described in 4.2 and K-Means from the OpenCV function *cv::kmeans*, which trains the classifier. The OpenCV implementation of K-Means defined in *modules/core/src/matrix.cpp* uses the square of the $l_2 - norm$, by the function *cv::normL2Sqr_* defined in *modules/core/src/stat.cpp*. This matches with the definition in 3.3.1. The feature space used in the K-Means implementation must be standardized as it uses the squared $l_2 - norm$, the Squared Euclidean Distance. Otherwise one dimension could cause other dimensions to be negligible, as discussed in 3.4.3. The feature space was standardized to be zero mean and unit standard deviation using the equations 3.49, 3.50, 3.51, 3.52. I.e. the vectors $\vec{\mu}, \vec{\sigma}$ were extracted from the entire training set. The means and standard deviation vectors were recorded and future data was standardized using the same factors. Predictions were created using equation 3.17.

The training set for textons included the entire image, i.e. the valid FoV given the cameras to include. This is because K-Means is unsupervised and does not require labels, as opposed the the classifier which does require labels.

4.4 Feature Implementation

Implementation details are included here if the information in 2 Literature Review and 3 Theory would not be sufficient in order to reproduce the feature. In general, all the features with the exception of *Saturation* over *Value* were rescaled into the representation of a uchar given the theoretical bounds for display purposes and to bin the inputs to a smaller feature space. See 4.4.3 for the explanation for this exception.

4.4.1 Normalized Difference Vegetation Index (NDVI)

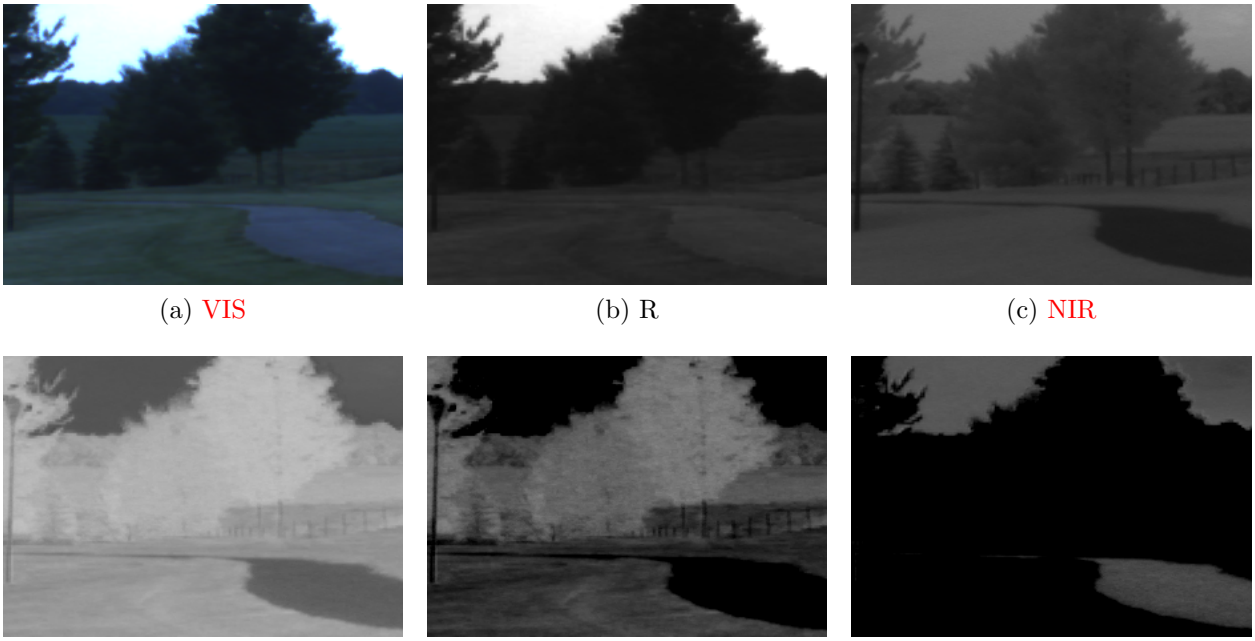
NDVI was pursued as a feature for vegetation detection given the simplicity and the demonstration by Salamati et al.[105] that the use of NIR and red together improved vegetation overall. Salamati et al. also observed however that confusion between different vegetation types increased when NIR was used.

The other options of MExG 2.12 and ExR 2.11 were not used for this iteration of this work. Feature selection optimization was not an objective for this study, so simply the most promising vegetation feature was chosen: NDVI. Note that the relationship of the normalized difference between the two bands are not describable through the RBF kernel, see sections 4.5.1,3.3.2.

Previous work in the Mechatronics Lab and preliminary results have shown NDVI to be effective at detecting vegetation, and have observed some key trends: 1) dry vegetation is drastically different than vegetation that is alive 2) some paints produces false positives 3) NDVI shows promise in detecting sky 4) vegetation responds to one side of the 0 in NDVI space depending on the order of the signs in the numerator of NDVI. The poorr response to dry vegetation follows from [106] that such targets have lower red absorption than greener targets. The false positives on paint is follows the findings of [3] which had high responses for some cars. The sky has also been seen to have a consistent response which also follows [3] that had observed the sky to have a distinct cluster in the NIR/R ratio. These preliminary findings showing the feature to have these patterns of responses make it desirable. The last item, that vegetation is almost always in the bounds of $[0, ..., 1]$ (for the formulation here) prompted the separation of the NDVI feature into an upper and a lower portion, equations 4.64.5. Note that the use of $\geq, <$ or $>, \leq$ is arbitrary as a floating point variable is highly unlikely to be exactly 0. The outputs were also rescaled to fit uchar representation, negating the negative bounds.

$$NDVI_{up} = \begin{cases} NDVI & \text{if } NDVI > 0 \\ 0 & \text{else} \end{cases} \tag{4.5}$$

$$NDVI_{low} = \begin{cases} NDVI & \text{if } NDVI \leq 0 \\ 0 & \text{else} \end{cases} \tag{4.6}$$

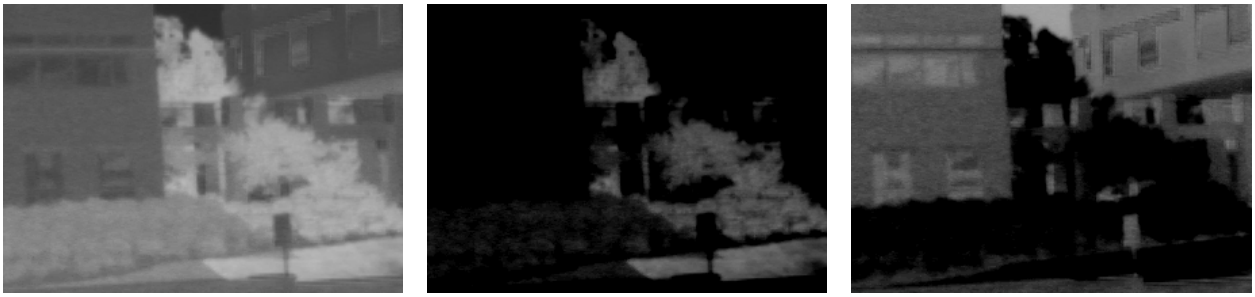


(a) VIS (b) R (c) NIR

(d) $NDVI$ (e) $NDVI_{up}$ (f) $NDVI_{low}$

Figure 4.4.1: Preliminary NDVI results for sky, canopy, grass, asphalt. The NDVI feature is split about 0.

(a) VIS (b) R (c) NIR



(d) $NDVI$ (e) $NDVI_{up}$ (f) $NDVI_{low}$

Figure 4.4.2: Preliminary NDVI results for a building in/out of shadow, canopy, grass, sky. The NDVI feature is split about 0.

Preliminary results are shown in figures 4.4.1 and 4.4.2. One can see that the differences are stark between the upper/lower images, as well as a finer texture due to equalizing a smaller bound over the uchar representation.

4.4.2 Normalized Difference Water Index (NDWI)

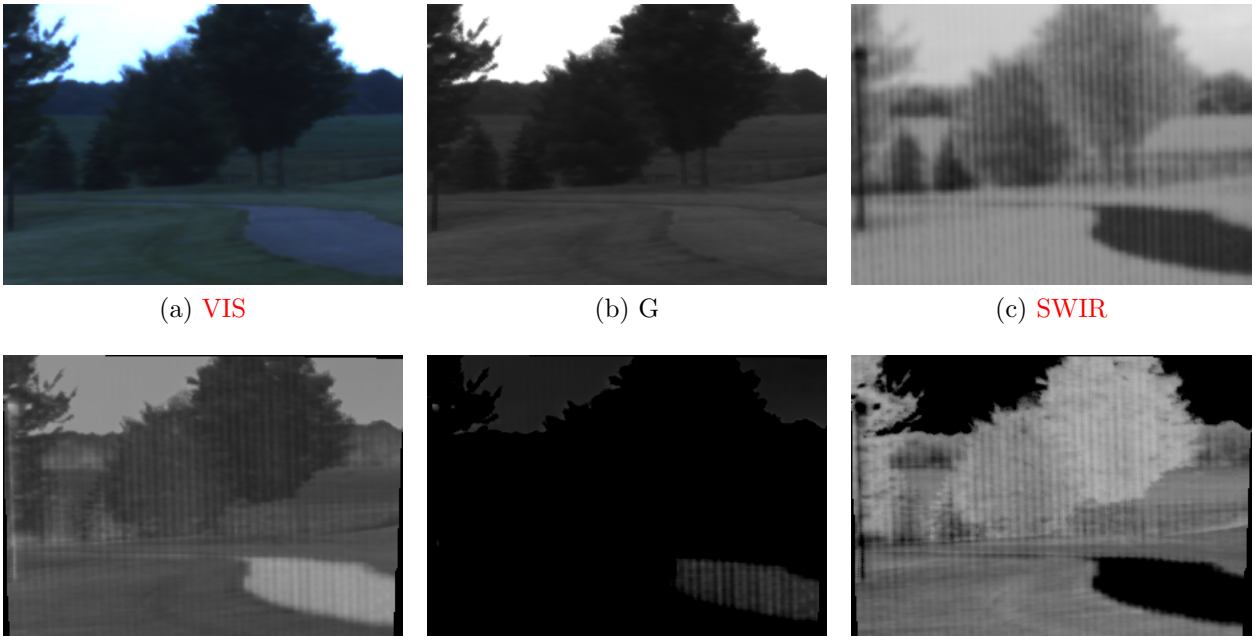
The formula chosen for the NDWI used is given in 4.7 and uses the green and SWIR bands: approximately $500-575nm$[22], and $900-1700nm$[23]. It is most similar to equation 2.6[68] in section 2.1.2. This was chosen due to the numerous statements in the literature that water generally absorbs highly above $1400nm$[66] and in NIR[9], and visible light highly reflects off of water surfaces[66]. The literature in general varied widely based on what the target was as well as the bands available, soil moisture vs. water body vs. vegetation moisture for example. This study was mainly interested in water bodies as opposed to the moisture content of soil which made the sources [68][66] most applicable.

It was required to choose a band that highly reflects off of water bodies as well as one that is highly absorbed. SWIR was chosen over NIR as "liquid water absorption in the $1500-2500nm$ is significantly stronger [than $900-1300nm$]"[66]. The SWIR band used here overlaps the former much more$(1500-1700)$ than the latter. The green band was chosen as

it is reflected highly off of bodies of water[66]. The choice of G over R, B, or L^*, was due to vegetation also reflecting G highly, but absorbing R and B. This should produce a lower response to vegetation as both G and SWIR reflect off of vegetation highly.

$$NDWI = \frac{G - SWIR}{G + SWIR} \tag{4.7}$$

The NDWI feature was split and rescaled about the 0 mark, similarly to the NDVI as explained in 4.4.1. Preliminary results of this feature are shown in figures 4.4.3 and 4.4.2. One can see than vegetation has a low response, and overexposure of the sky and its reflection are high. The preliminary results of this feature show it to be similar to NDVI. The vertical line banding artifact seen in the SWIR band of 4.4.3 is due to sensor noise.



(a) VIS (b) G (c) SWIR

(d) $NDWI$ (e) $NDWI_{up}$ (f) $NDWI_{low}$

Figure 4.4.3: Preliminary NDWI results for canopy, grass, asphalt, sky. The NDWI feature is split about 0.

(a) VIS (b) G (c) SWIR

(d) $NDWI$ (e) $NDWI_{up}$ (f) $NDWI_{low}$

Figure 4.4.4: Preliminary NDWI results for canopy, overexposed water reflection, water reflection, sky. The NDWI feature is split about 0.

4.4.3 Saturation over Value

The calculation for S/V is straightforward but the representational space was displaced to deal with the possibility of dividing by zero when dealing with images of type uchar ($[0, ..., 255]$). First the implementation of OpenCV cv::cvtColor was used to transform to the VIS data to HSV space, which uses equation equation 4.8[39]. Note that the bounds given for the HSV planes are theoretical, it is scaled depending on the encoding precision in application. In this particular case the output was of uchar, so the bounds were $V \in [0, ..., 255]$ $S \in [0, ..., 255]$ $H \in [0, ..., 360/2]$. Equation 4.9 was then applied with a destination encoding of a float. The non-integer destination was necessary as there is a significant region of the output space that is small. In addition, rescaling the output bounds of $[1/256, ..., 256]$ to uchar representation was not pursued as there was not enough precision to encode the majority of the values in preliminary tests. Note that if one plots $[0.99, 0.98, ..., 0.01]$ against $[0.01, 0.02, ..., 0.99]$ the curve is exponential. This is acceptable as the feature standardization step will displace and scale in order to produce zero mean and standard deviation, see 3.4.3.

$$
\begin{aligned}
V &\leftarrow max(R,G,B) \\
S &\leftarrow \begin{cases} (V - min(R,G,B))/(V) & \text{if } V \neq 0 \\ 0 & \text{else} \end{cases} \\
H &\leftarrow \begin{cases} 60(G-B)/(V - min(R,G,B)) & \text{if } V_{max} = R \\ 120 + 60(B-R)/(V - min(R,G,B)) & \text{if } V_{max} = G \\ 240 + 60(R-G)/(V - min(R,G,B)) & \text{if } V_{max} = B \end{cases}
\end{aligned} \tag{4.8}
$$

where

$R = Red\ Intensity$ $G = Green\ Intensity$ $B = Blue\ Intensity$

$H = Hue \in [0, ..., 360]$ $S = Saturation \in [0, ..., 1]$ $V = Value \in [0, ..., 1]$

$$\text{Saturation over Value} = \frac{S+1}{V+1}; \tag{4.9}$$

4.5 SVM Implementation

The SVM code of the OpenCV library was used to perform supervised classification for this system, of which is "based on LibSVM[39]. The available SVM library can handle multi-class classification, multiple regularization formulations including C-SVM, and four built in kernels: linear, polynomial, RBF, and sigmoid.

4.5.1 SVM Kernel Selection

Chih-wei Hsu, Chih-chung Chang, and Chih-jen Lin, of LibSVM recommend the RBF kernel as a first choice for three primary reasons (1) the RBF is nonlinear and so will be able to handle more complex decision surfaces (2) the polynomial kernel has an additional parameter for the number of polynomials which complicates the cross validation process (3) the RBF kernel "has fewer numerical difficulties" than the polynomial and sigmoid kernel[97]. They note that if the number of features are much larger than the number of instances than the linear kernel may have a sufficiently large kernel space. Higher mapping is often used in the opposite case where the number of features is small.

The RBF kernel was chosen as the first kernel to use given that the number of features of this system is small, order of 10^1, compared to the number of instances, order of 10^6. While the authors note that their LIBLINEAR package could perform more quickly given large

amounts of data, the RBF kernel will perform just as good as the linear after a parameter search as the linear kernel is a special case of the RBF kernel.

4.5.2 SVM Parameter Optimization with Grid Search

Chih-wei Hsu, Chih-chung Chang, and Chih-jen Lin, of LibSVM recommend performing a grid search strategy for parameter selection[97]. Their reasoning for this is that while some more advanced heuristic methods could be used a simpler grid search can be faster given the simplicity of their recommended formulation: C-SVM with the RBF kernel, which requires 2 parameters C, γ. Additionally, each point in the grid search can be parallelized to decrease the computation time. The grid search is performed twice, first for a coarse span of parameters second with a finer scale. It is performed as follows: select bounds for the model parameters, perform cross validation and average the validation accuracy for each parameter combination, select the best performance and repeat for a finer grid scale. The authors recommend that a smaller fraction of the training set can be used during the coarse grid step if the number of samples are large. The authors also recommend performing the grid search on a logarithmic scale, such as $C = [2^{-5}, 2^{-3}, ..., 2^{15}]$ as they consider it more practical.

A logarithmic scale was used to search for C, γ. First a coarse scale was used to find the best parameters over large bounds. A second search over a finer scale was centered on the best parameters found in the coarse step. K fold cross validation was used with 5 folds and shuffled data. The coarse cross validation used a fraction of the training data, 0.20, in order to decrease computation time while the fine cross validation used the entire training set. This was done for practical reasons as preliminary tests using all the data available were projected to take days to train, but also under the recommendation from the authors of LibSVM: "[The coarse/fine cross validation grid search] works well for problems with thousands or more data points. For [large sets] a feasible approach is to randomly choose a subset [for the coarse step and use the complete set for the fine grid]"[97]. Also note that Hsu, Chang, and Lin recommend "the [grid search] procedure works well for data which have [on the order of 10^1 or 10^2] features", while feature sets on the order of 10^3 may require feature selection[97]. This implementation uses a feature set on the order of 10^1.

This implementation optimized the *F1-Weighted Average* score 3.48 and implemented class weights, in order to account for class imbalance. The total misclassification rate was not used as it "may not be a relevant choice [for multi-class problems]"[107]. Therefore the *F1-Weighted Average* was used as it weighted the scores given the percentage that the class was represented in the set, contra to the macro average which places equal weight on each class. The class weights of the OpenCV implementation are contained in *CvSVM-Params::class_weights*, which "are multiplied by C so the parameter C of class $\#i$ becomes $class\_weights\_i * C$... The larger weight, the larger penalty on misclassification of data from the corresponding class"[39]. The class weights were chosen using the same concept of equa-

tion 3.48, where the weighted average was computed using the class percentages. However, the percentages were inverted as the class weights in the C-SVM problem are the penalty value for misclassification. The class that has fewer instances must be given more weight and the larger instance class must be given less weight. This is the concept used by [100]. The class weights were recalculated for call of the *CvSVM::train(...)*, and the equation for this is seen in 4.10. Note that the class weights were the normalized inverted fraction of the class instances with respect to the total number of labels.

Further details on the bounds and other hyperparameters of the grid search step can be found in 5.1 Hyperparameters.

$$\begin{aligned} N &= \sum_{c=0}^{\eta-1} N_+(c) \\ C_i &= \frac{\frac{N}{N_+(c)}}{\sum_{c=0}^{\eta-1} \frac{N}{N_+(c)}} \end{aligned} \tag{4.10}$$

where

$$\eta = \#\text{ of classes} \qquad c \in [0, 1, \ldots, \eta - 1] \qquad N_+(c) = \#\text{ instances of } c$$

This implementation multi-threaded the CV step using *QtConcurrent*[108] as the grid search iterations are independent. This optimization is also recommended by Hsu, Chang, and Lin[97]. The *QtConcurrent* class allows for the straightforward implementation of multi-threaded applications and "automatically adjust[s] the number of threads used according to the number of processor cores available"[108]. The use of this class resulted in full utilization of the available CPU cores, compared to approximately less than 75% utilization without.

4.6 Data Set Creation

The training and testing data used in this work was labeled on the pixel level. The pixel level was pursued because previous work in the Mechatronics Lab was able to most quickly integrate pixel level data into an existing system, discussed in Motivation 1.1. It was also pursued as pixel level features are relatively straightforward to create. The disadvantage is that the hand labeling process is more difficult when one requires pixel level ground truths as opposed to simply tagging and image with a keyword or drawing a box around the target. This section addresses simplifications made to the labeling strategy, the tool used to encode the hand labels, the selection of classes, and the practical accommodations taken when handling the data.

4.6.1 Simplifications in the Labeling Process

One significant difficulty in hand labeling images is reducing label noise, i.e mislabeled ground truths. This can easily occur on edges which can be difficult for a user to segment. Zooming in on an image will reveal that the edges of targets can often blend in easily with the backgrounds, and such regions can be ambiguous and/or highly variegated. Labeling in binary values exacerbates label noise as the use must decide either true or false even in these ambiguous regions. The PASCAL VOC database used a void class in order to reduce this problem seen in the off-white color of figure 1.3.7. The void class identified pixels that were not used for testing or training and was used to "mask out ambiguous, difficult or heavily occluded objects...[including objects] too small to be marked"[109]. Objects not in the class list, void inclusive, of the PASCAL VOC database were encoded as the background class using black pixels.

The decision was made early on for the purposes of this study to make use of the void class heavily but not to include the background class; the goal being to reduce the burden on the user producing the label. This is because many objects in the scenes available to this study are often porous or thin, such as tree canopies or tall stalks of grass. Labeling such imagery would require liberal use of the void class in order to avoid label noise in these regions. The result is a system where one labels only the positive samples of a class and only the areas that can be accurately labeled. Recall that in the *one vs. all* multi-class classification strategy described in 3.3.3 that all other samples labeled as a different class are counted as a negative sample for the class at hand. This strategy was used by Namin and Petersson [1] where the hand labels took up a relatively small portion of the entire image.

4.6.2 Labeling Tool

The encoding strategy chosen to store annotations in this study defines the classes as specific colors in an accompanying RGB image, similar to the PASCAL VOC database[109] and Namin and Petersson[1]. The RGB class image is registered to the VIS/NIR images in this case. This was chosen as compatibility with on-line labeling tools was not a requirement of this work, and, the data used in this work was unique regarding the number of bands used. Being able to inspect the additional bands and the registration results was beneficial in the labeling process. LabelME for example is a web based tool that allows users to upload data and store the annotations in XML files[84]. Objects are outlined coarsely by straight lines and the vertices of the polygons are stored along with a class. Storing and parsing the class information in RGB colors was preferred as it would allow finer segmentation and the code required to parse the data would also be used in the other phases of the project. A full discussion of annotation methodologies is outside the scope of this project, so described here are the capabilities developed to process and label the data.

A GUI was developed using the Qt C++ library[108] and OpenCV library[39]. The primary

components of the GUI were a QMainWindow and QLabel for interacting with an image, QDockWidget objects for showing a list of the available images as icons, and a QToolBox for showing the different stages of processing. The main capabilities are:

- Allow the user to view the imagery in two lists of icons: one with respect to one camera over time, and one with respect to all available cameras for one frame (time stamp). Allow the user to select these images for processing & display.
- Allow the user to view the results of image registration.
- Allow the user to define sets of images, e.g. a list of timestamps associated with a test/train partition. Allow user to view registration results during this interface.
- Allow the user to define a set of target classes.
- Allow the user view the hand labeled class image overlaid atop any of the source imagery and to modify the hand label through a mouse. Allow the user to zoom, pan, and scroll over the display.
- Allow the user to push experiments to a thread, save and load the results.
- Allow the user to view the results of an experiment and save images of the features and predicted classes.

4.6.3 Class Selection

The classes chosen were modeled after Namin and Petersson, where classes were included if those targets were "readily marked up"[1]. Such classes were also separated into in and out of shadow if such a designation was feasible. The full list including the colors used to encode the hand labels are shown in table 4.2. Vegetation was separated based on the general principle of weather an all terrain vehicle could drive over said vegetation, as the original motivations for this work were to identify im/passable vegetation. Large bushes and tree canopies for example were identified as impassable while cut and tall grass were identified as passable. Meadows were considered as passable, while crops were ambiguous and not labeled. The *Sky* and *Clouds in Sky* classes were separated from the parent class of *All Sky* as the signatures differed greatly; likewise for *Bark/Cut Wood* and *Reflections on Water/Water*. *Vehicles* and *Glass* were combined in order make labeling easier. *Rock* and *Brick* were included as there were many buildings made of Hokie Stone bricks, a type of limestone common to the Virginia Tech Campus, but not many instances of boulders. Preliminary results showed that many buildings were confused with vehicles, so the *Brick & Rock* class was added. Reflections were a prime concern and so water was separated into regions where the reflections allowed one to discern the reflected object and those that this could not be done. This was termed *Reflections on Water* and *Water* respectively.

It should be noted that a comprehensive optimization of the class list was outside the scope of this project. Again, a class was included if it was "readily marked up"[1]. Classes that could be functionally the same were separated if the appearance was remarkably different,

primarily water reflections and clouds vs. sky. Conclusions regarding the class selection are discussed in 7 Future Work.

Table 4.2: List of classes for this system

No.	Color	Target	Description
0		Asphalt	Roadways, paths, etc.
1		Asphalt in Shadow	Roadways, paths, etc. in shadow of any object
2		Grass	Cut grass, vegetation that can be driven over
3		Grass in Shadow	Cut grass, in shadow of any object
4		Bush and Canopies	Tree canopies and bushes that cannot be driven over
5		Concrete	All concrete, including in shadow
6		Sky	Regions of the sky that are unobstructed
7		Clouds in Sky	Regions of the sky obstructed by clouds
8		Water	Water where the surface is disturbed or the pool bottom is visible or turbid
9		Reflections in Water	Water where objects are visible reflected on the surface
10		Bark	Exterior of trees
11		Cut Wood	Wood that is bare and is not covered by paint, e.g. light poles
12		Vehicle & Glass	Cars and windows
14		Rock & Brick	Bare rock and bricks of buildings, boulders

4.6.4 Training/Test Set Partitioning

The available data was partitioned into a test and a training set for the system as a whole. The test set was consistent throughout experiments and was used to produce the metrics for evaluation, namely the *F1 Weighted-Average* 3.4.2 F-Scores. The train/test split was performed randomly on a per image basis, i.e. out of the 133 registered frames available a percentage 0.60 frames were chosen for training and the remainder 0.40 were chosen for testing. The frames assigned for testing or training were split randomly once and maintained between experiments, rather than splitting randomly each time. The split was performed on a per image basis as opposed to a per sample basis where the feature matrix for all images would be built, randomized, then split. Splitting the data on a sample basis would result in a test set that would not sufficiently represent new data. The samples within the frames are highly correlated with one another, especially with respect to lighting. Preliminary experiments showed that test accuracies were consistently high, close to 99%, when split on a sample basis which follows the idea that this method is not sufficient.

4.6.5 Data Subsets

The data available to this study was 5.024×10^6 labeled pixels from 133 images. The practical considerations that were taken to address this amount of data are described here while the larger, and ongoing, question of how much data is needed to perform a task is discussed further in 5.3 Percent Training Data Convergence and 7.1.2 Data Collection and Data Set Size.

The main practical consideration is training time as it is generally proportional to data set size. Preliminary experiments showed that developing on a full dataset would be inhibitory as such tests would require time on the order of days or weeks. Subsets of the data were used in order to develop more quickly, to provide insight into the learning curve, and to perform convergence testing on hyper-parameters: K convergence of section 5.4 for example. The subsets of the data were calculated in the same way that the test/train split was, which is on a per image basis. A random subset of available pixels from each training image was taken and appended to the conglomerate train list. The test set was not reduced in order to maintain consistent standard. The data for K-Means was also not reduced, as the training time for K-Means was much smaller than the training of the SVM, it did not require labels, and to simplify the programming task.

4.7 Feature List

The features used by this study are summarized in table 4.3 for a total of 18 features. Note that for implementation purposes scaling and displacement was performed on the imagery prior to feature standardization. All images with the exception of S/V were converted to single channel uchar, i.e. CV_8UC1 of OpenCV[39]. Also, the normalized difference indices were split about the 0 point for an upper and lower image. See 4.4 for the details.

Table 4.3: Imagery used as Features

Source	Representation	Theoretical Bounds
NIR	uchar	-
SWIR	uchar	-
$CIE\ L^*a^*b^*\ L*$	uchar	$[0, ..., 100]$[49]
$CIE\ L^*a^*b^*\ a*$	uchar	$\pm 100]$[49]
$CIE\ L^*a^*b^*\ b*$	uchar	$\pm 100]$[49]
DoLP	uchar	$[0, ..., 1]$
Polarization Phase	uchar	$[0, ..., 360]$
S_0	uchar	$[0, ..., 255 * 2]$(uchar inputs)
S_1	uchar	± 255 (uchar inputs)
S_2	uchar	$[-255 * 2, ..., 255 * 4]$ (uchar inputs)
S_3	uchar	$[-255 * 2, ..., 255 * 4]$ (uchar inputs)
$NDVI_{up}$	uchar	$[0, ..., 1]$
$NDVI_{low}$	uchar	$[-1, ..., 0]$
$NDWI_{up}$	uchar	$[0, ..., 1]$
$NDWI_{low}$	uchar	$[-1, ..., 0]$
S/V	float	$[0, ..., \infty]$
Texton	unsigned integer	$[0, ..., K_{textons} - 1]$

Chapter 5

Experiments

5.1 Hyperparameters

The hyperparameters listing in this section were common to each test.

5.1.1 Cross Validation Hyperparameters

The grid search for the SVM hyperparameters was done logarithmically as discussed in 4.5.2 SVM Parameter Optimization with Grid Search. The input bounds for such said grids were chosen through trial and error of preliminary tests: the grid search should be sufficiently large to show convergence, but also be limited in size to decrease training time. The chosen grids were generous in size at the expense of precision and as such there were often clear borders of low performing regions. The general case for computing the grid is shown in equations 5.1, and further detailed in equations 5.2 to 5.4.

Put simply, 8 samples for C and γ were used for the coarse step and 5 samples for C and γ were used for the fine step which was centered on the maximum score from the coarse step with half the delta on either side of the said maximum.

$$\begin{aligned} exp_i(i) &= exp_{min} + \delta_{exp} * i & i &\in [0, 1, ..., ((N+2)-1)] \\ N_{samples} &= N + 2 & \delta_{exp} &= (exp_{max} - exp_{min})/((N+2)-1) \end{aligned} \tag{5.1}$$

where

$$N > 0$$

The coarse grid inputs are

$$\begin{aligned} N_{samples} &= 8 \\ (\log_2 C) &= exp_C \in [0, ..., 18] \\ (\log_2 \gamma) &= exp_\gamma \in [-9, ...9] \end{aligned} \tag{5.2}$$

which results in

$$\delta_{expC} = 3 \qquad \delta_{exp\gamma} = 3$$

The fine grid inputs are:

$$\begin{aligned} (\log_2 C)_{min} &= (log_2 C)^{coarse}_{F1max} - \delta_C/2 \quad (\log_2 C)_{max} = (log_2 C)^{coarse}_{F1max} + \delta_C/2 \\ (\log_2 \gamma)_{min} &= (log_2 \gamma)^{coarse}_{F1max} - \delta_\gamma/2 \quad (\log_2 \gamma)_{max} = (log_2 \gamma)^{coarse}_{F1max} + \delta_\gamma/2 \\ N_{samples} &= 3 \end{aligned} \tag{5.3}$$

where

$(log_2 C)^{coarse}_{F1max}$ is the C value where F1 Weighted is maximum for the coarse step
$(log_2 \gamma)^{coarse}_{F1max}$ is the γ value where F1 Weighted is maximum for the coarse step

The additional parameter of number of folds was

$$N_{folds} = 5 \tag{5.4}$$

which was chosen as the examples in [97] used this value.

The percentage of training data used in the coarse cross validation was lower than 1, as discussed in 4.5.2. The percentages used were

$$P_{coarse} = 0.20 \quad P_{fine} = 1.0 \tag{5.5}$$

5.1.2 Test/Train Split Hyperparameters

The 133 available frames were split randomly into 0.60 train and 0.40 test. In general the larger the training set size the more confident one can be in the result, but since the optimization of this parameter is outside the scope of this study it was chosen somewhat arbitrarily. Note that [14] split their television data set by 0.50, which they believed produced a "reasonably independent" train and test set.

5.1.3 Texton Hyperparameters

The base sigma size k was chosen as 1.0. Again, this was somewhat arbitrary as converging this hyperparameter was outside the scope if this study. Note that Shotton, Winn, Rother, and Criminisi[14] used $k = 0.7$ for their Microsoft Research Cambridge (MSRC) and television combined data set.

The filter bank used in this study was the same as Shotton et al. as discussed in 4.2. Shotton et al. applied the Gaussian filters to all three channels of the $CIE\ L^*a^*b^*$ space, while the other filters were applied only to the $a*$ and $b*$, shown in 4.1. This study however had more than the three channels of $CIE\ L^*a^*b^*$ available. The apparent trend from Shotton et al. was that Gaussian blurs were applied to all sources encoding color while the additional filters were applied to the sources encoding texture. Note that the $a*, b*$ planes are meant to be free from lighting affects. The decision was made for this study to apply the Gaussian blurs to all imagery and to reserve the remaining filters for images that appeared to have unique texture or response. The NDWI images for example clearly displayed a reduced texture upon reflection compared to the other difference index NDVI so all filters were applied. Similarly it was apparent from the preliminary imagery as well as the literature[11] that low textured regions of the S/V space were common to the sky and reflection pixels and all filters were applied to these sources as well.

Parameter sweeps regarding the application of filters was outside the scope of this study and were not changed between tests.

5.2 Common Evaluation Methods

The methods used to evaluate classifiers without regard to any convergence tests were:

- Confusion matrix
- Plot of the number of samples vs. class
- Plot of the F1-Weighted Average score vs. class

The confusion matrix provides quantitative insight into which classes are confused, or misclassified, with other classes. This is explained in 3.4.2 Confusion Matrix and its Derivatives. Sky pixels for example are often misclassified as reflections or water as the color and saturation are very similar.

Plotting the number of samples vs. class index can show the distribution of the samples available to the classifier. This is important as multi class problems can suffer from the *class imbalance* problem, where a "classifier trained on an imbalanced dataset can produce suboptimal models which are biased towards the majority class"[110]. Namin & Petersson

Table 5.1: Filter Application to Feature Map from Filter Bank of Table 4.1

Feature	Filters Applied
NIR	All Filters
SWIR	All Filters
$CIE\ L^*a^*b^*\ L*$	All Filters
$CIE\ L^*a^*b^*\ a*$	Gaussian $k, 2k, 4k$
$CIE\ L^*a^*b^*\ b*$	Gaussian $k, 2k, 4k$
DoLP	All Filters
Polarization Phase	Gaussian $k, 2k, 4k$
S_0	Gaussian $k, 2k, 4k$
S_1	Gaussian $k, 2k, 4k$
S_2	Gaussian $k, 2k, 4k$
S_3	Gaussian $k, 2k, 4k$
$NDVI_{up}$	Gaussian $k, 2k, 4k$
$NDVI_{low}$	Gaussian $k, 2k, 4k$
$NDWI_{up}$	All Filters
$NDWI_{low}$	All Filters
S/V	All Filters
Texton	N/A

for example randomly sampled $15,000$ pixels from each class so that the distribution was consistent[1].

Plotting the F1-Weighted Average allows one to inspect which classes had the most success/lowest performance.

5.3 Percent Training Data Convergence

Supervised machine learning has a quintessential question: "what is the expected classification performance for a given training sample size?"[90]. Producing labeled data can often take large amounts of time and money, and knowing how much data is required to perform a certain task can be very useful when evaluating the feasibility and success of such a machine learning application. The concept of a learning curve from [90] is used here in order to address this question, which is discussed in 3.4.2 Learning Curves & Convergence. A success metric is plotted versus the input training set size with the general expectation that the classifier will converge to a maximum performance as the training size increases. Active learning and the prediction of the required data set size as discussed in [90] is outside the scope of this study.

One rule of thumb regarding the required data set size is: "The number of independent

training samples needed per class is greater than or equal to five times the number of features used" [21]. While there are 5.024×10^6 labeled pixels available to this study, it should be noted that not every pixel is independent. Pixels can imply important information regarding the adjacent pixels; this is the premise for the use of context in the image labeling problem. It might be more accurate to say that there are a certain number of independent regions within an image. The lower limit would be one region per frame, resulting in a required 1170 frames for 13 classes and a maximum 18 features used in this data set. There are certainly more than 1 independent region per frame in this dataset however.

The goal of the convergence tests is to find if the classifier converges, and if so, the amount of data required to do so. The test set size is constant throughout this parameter sweep, as discussed in 4.6.5 Data Subsets.

The methods used to evaluate the percent data convergence tests are:

- Plot *F1 Weighted-Average* vs. C/γ for the fine and coarse cross validation.
- Plot percent training data vs. *F1 Weighted-Average*.
- Plot class index vs. number of samples per class.
- Plot class index vs. *F1*.

5.4 K Centers Convergence

The goal of the parameter sweep for the number of textons, K, is to find the optimum value with respect to the performance metric *F1 Weighted-Average*. Shotton, Whinn, Rother, and Criminisi performed a parameter sweep for K using the approximate values $25, 50, 200, 300, 400$[14]. Shotton et al. found that there was an initial and drastic improvement from 25 to 50, which leveled off and began to decrease after the 200 value, where the system began to "overfit" as K increased. This study maintains the logarithmic parameter sweep used in the cross validation steps and tested $K \in [4, 8, ..., 1024]$.

The methods used to evaluate the K convergence tests are:

- Plot *F1 Weighted-Average* vs. $C\&\gamma$ for the fine and coarse cross validation.
- Plot K vs. *F1 Weighted-Average*.
- Plot class index vs. number of samples per class.
- Plot class index vs. *F1*.

Chapter 6

Results & Conclusions

6.1 Data Convergence

Figure 6.1.1 shows the convergence of the percent training data used, where the F1-Weighted Average score is plotted versus the size of the training set. The number of texton centers was $K = 128$, the percentage was swept logarithmically, and the pixel counts were: 1.919×10^6 test/ 3.105×10^6 train. The purpose of this test was to find the required amount of training data to reach a certain performance. A learning curve typically converges to a maximum performance at which no further training data is beneficial. This however is not seen for the experiment. The maximum performance of $F1_W = 0.623$ was achieved using the smallest number of training samples: 1% training data. The lowest score was $F1_W = 0.468$ at 28.5%, and the score using 100% of the data was $F1_W = 0.516$. The first question is why does it not converge at higher sample counts? The performance actually drops as more data is included, albeit with a brief spike in the performance at 65.8% which then drops once more at 100%. One possible answer is that the training samples used to train on are imbalanced, and has resulted in a majority classifier. To explore this possibility, one can plot the number of samples versus the class: figure 6.1.2.

The plot suggests that there is indeed a large variance. $\sigma = 3.591 * 10^3$ for the 1% sample, and $\sigma = 359.6 * 10^3$ for the 100% sample showing an increase of an order of 2. This does not include the null *Human* class of which data was not sufficiently available and therefore not included. Note that the lack of the human class does not affect the $F1_W$ score as the weight is 0. As a reference there is an average of $221.8 * 10^3$ samples per class in the entirety of the data set. The samples vs. class for the test set is also shown in figure 6.1.3. The class imbalance can lead to a majority classifier where there is a high bias towards the largest class, and the smaller classes do not perform as well.

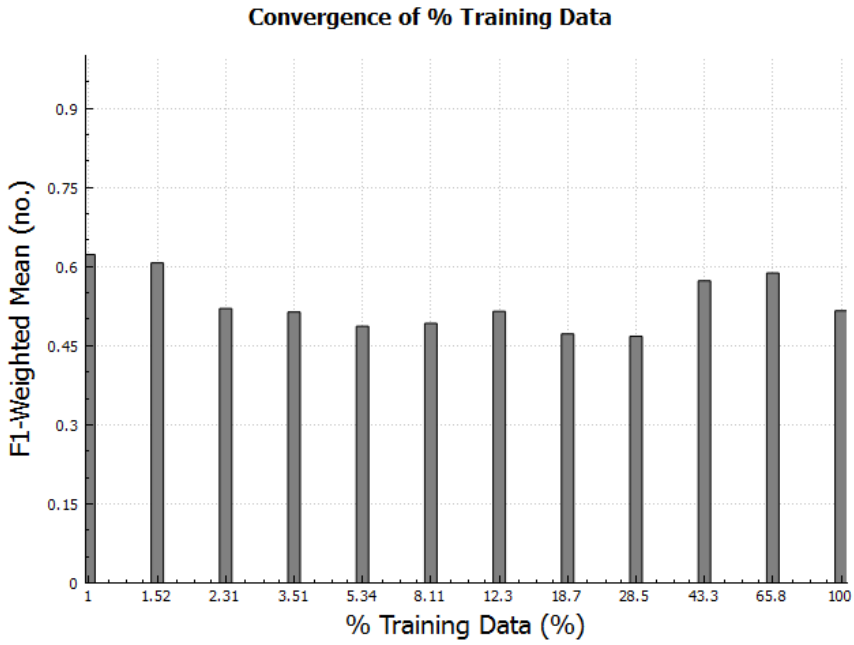


Figure 6.1.1: Convergence plot of the percent training data used versus F1-Weighted Average of the test set.

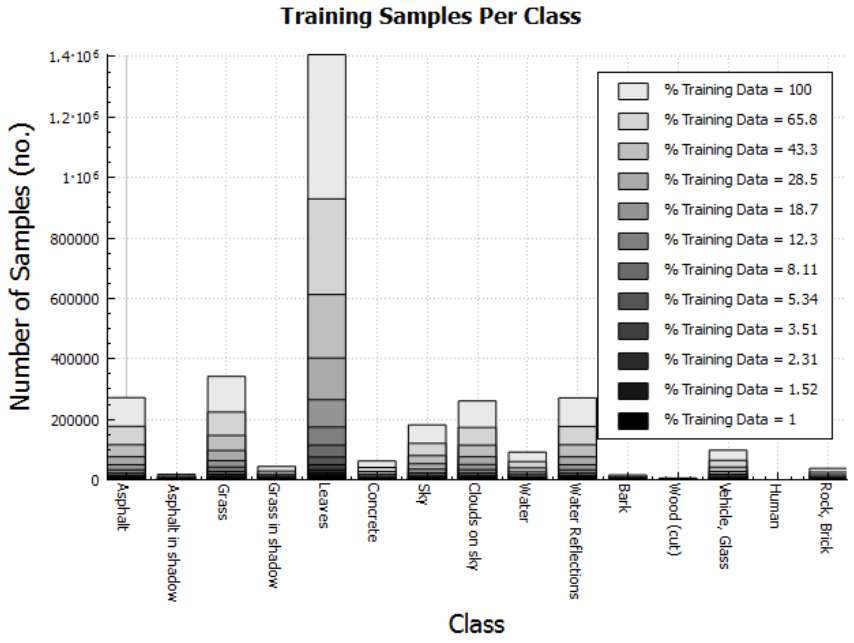


Figure 6.1.2: Convergence plot of the percent training data used versus number of training samples per class.

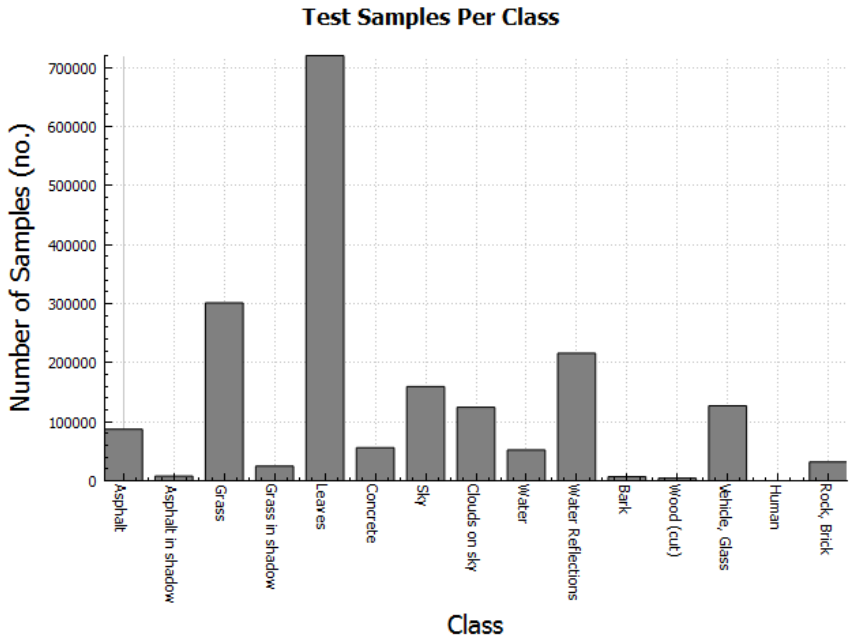


Figure 6.1.3: Number of test samples per class. The test set was consistent through all experiments so there is no convergence of any variable.

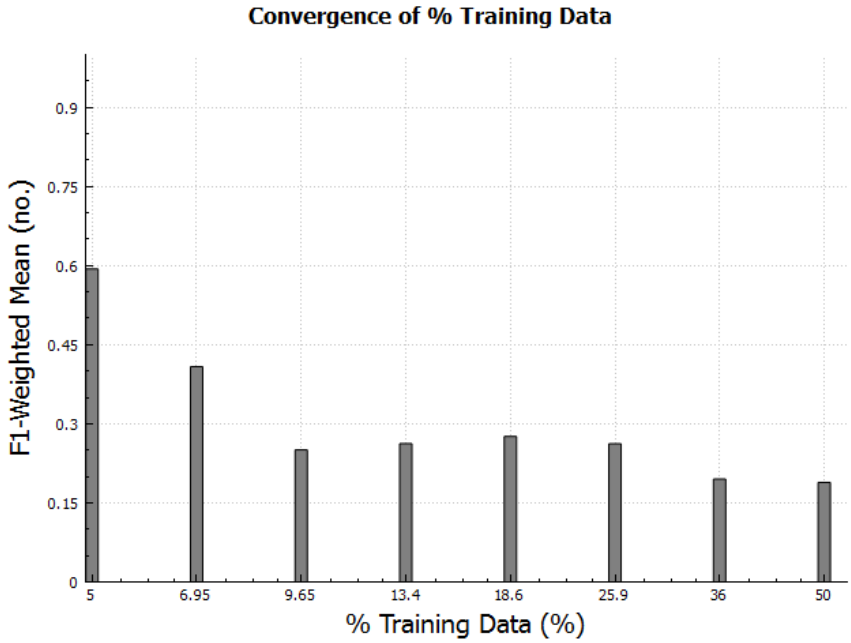


Figure 6.1.4: Exploratory test for oversampling, $F1_W$ vs. % training data. Note that the ending score is much lower and falls more rapidly than without the oversampling.

While not previously discussed, one method to combat the class imbalance problem is over-sampling or under-sampling[111]. Over-sampling approximates or copies samples to boost the number available. Under-sampling disregards samples in order to decrease the number. In one preliminary test over-sampling of all the samples available was performed inadvertently but the result gives insight into the problem. Figure 6.1.4 shows this result, which shows a similar score for the 1% trial, but a much steeper drop and lower final score. In this preliminary test the larger classes accumulated samples at a faster rate than the smaller ones, a linear convergence. This implies that the class imbalance problem can be an issue as the decrease in performance resulted from an exacerbation of the imbalance.

The confusion matrix may afford insight to the imbalance problem or the system as a whole by allowing one to inspect which classes are problematic. Tables 6.1 and 6.2 show the confusion matrices for the best and worst performing classifier respectively. Asphalt in shadow is predominately predicted as a vehicle or glass. Qualitatively speaking, this could be due to overexposure and similarities of shadowed asphalt in the polarimetric regions, see 1.3.2. Grass is often predicted as leaves, which makes sense intuitively as the two are similar in color and texture. Grass in shadow is also often confused with leaves which makes sense because leaves, a.k.a impassable vegetation, are very difficult to hand label separately from leaves in shadow. Therefore leaves and leaves in shadow are considered one class rather than two. Sky is often predicted as asphalt and clouds. Both are likely confused due to overexposure, and similarities in texture. However, the intensity of the SWIR is drastically different for these targets. Clouds are also similarly predicted as sky, likely for similar reasons for the sky/ asphalt/ cloud predictions. Water is often misclassified as glass and vehicles which could be heavily influenced by the polarization features: both polarize light well. Water reflections are misclassified as leaves, which is likely due to a high number of samples where vegetation was reflected. Since this study used primarily intensities, it makes sense intuitively that reflections of color are confused with similar colors. The polarametric information may combat this problem however, as fine texture is lost upon reflection and the degree of polarization increases. Bark is predicted predominately as leaves. This is likely influenced by the difficulty in labeling this class as well as the similar surrounding textures. The number of samples for this class however is very low, and it is confused with the largest class. It is likely heavily influenced by the class imbalance problem, as well as all of the classes with very few samples: asphalt in shadow, grass in shadow, bark, and cut wood.

Table 6.1: Confusion matrix for percent data conversion, $data = 1\%$. Row normalized by the number of positive samples, reported as percent (%). Rows: truth, Columns: predicted.

	Asphalt	Asphalt in shadow	Grass	Grass in shadow	Leaves	Concrete	Sky	Clouds on sky	Water	Water Reflections	Bark	Wood (cut)	Vehicle, Glass	Human	Rock, Brick
Asphalt	68.2	0	0	0	20.2	6	0.1	0	0.3	0.1	0	0	5	0	0
Asphalt in shadow	2.2	19.3	0.1	0.1	17	3.2	0	0	0.3	0.4	0	0	56.5	0	0.7
Grass	0	0	49.7	0.3	47.4	0	0	0.4	0.1	2.1	0	0	0	0	0
Grass in shadow	0	0	16.8	16	49.9	0	0	0	0	17.1	0	0	0.1	0	0
Leaves	0	0	5.6	0.7	90.5	0	0	0.1	0.1	2.8	0	0	0.1	0	0
Concrete	62.7	0.2	0.1	0	9.2	9.4	7.7	1	0.4	0.3	0	0	8.1	0	0.9
Sky	20.8	0	0	0	17.5	0.4	26.7	20.3	0.2	0	0	0	14	0	0
Clouds on sky	5	0	0	0	2.3	2.6	27.1	59.1	0.8	0	0	0	3.1	0	0
Water	3.3	0	0	0	13.9	0	0	0.1	34.1	9	0	0	39.6	0	0
Water Reflections	1.7	0	1.9	0.6	32.7	0.1	0.1	0.7	5	52.7	0	0	4.4	0	0.1
Bark	0.5	0	1.5	5.6	66.9	2.2	0	0	0.5	12.7	2.1	0	6.8	0	1.1
Wood (cut)	0.4	0	27.7	0	35.8	2.2	3.5	6.6	0.8	10.7	0	0	11	0	1.2
Vehicle, Glass	3.8	2.1	0	0	12.1	2.9	0.5	0.5	0.6	0.2	0.1	0	76.2	0	1.1
Human	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Rock, Brick	1.8	0.1	0.2	0	9.2	0.3	0.8	2.8	0	0.4	0	0	10.5	0	73.8

Table 6.2: Confusion matrix for percent data conversion, $data = 28.5\%$. Row normalized by the number of positive samples, reported as percent (%). Rows: truth, Columns: predicted.

	Asphalt	Asphalt in shadow	Grass	Grass in shadow	Leaves	Concrete	Sky	Clouds on sky	Water	Water Reflections	Bark	Wood (cut)	Vehicle, Glass	Human	Rock, Brick
Asphalt	71.7	0	0	0	13.8	5.3	0	0	0.2	0	0	0	8.2	0	0.8
Asphalt in shadow	3.1	0.1	0.1	0.2	15.3	1.4	0	0	0.4	0.1	0	0	78.8	0	0.7
Grass	0.1	0	15.8	0	83.2	0.1	0	0.3	0.1	0.3	0	0	0.1	0	0
Grass in shadow	0	0	8.4	8	78.7	0	0	0	0	4.4	0	0	0.5	0	0
Leaves	0	0	3.4	0.1	94.3	0	0	0.1	0.1	1.8	0	0	0.2	0	0
Concrete	57.6	0.3	0	0	9.5	15	2.6	1.2	0.2	0.2	0	0	11.8	0	1.6
Sky	20.4	0	0.1	0	20	1	8.9	18.2	0	0	0	0	31.5	0	0
Clouds on sky	4.9	0	0	0	5.7	3.2	22.9	50.8	0.3	0	0	0	12.2	0	0
Water	1.1	0	0	0	19.2	0	0	0	38.4	4.1	0	0	37.1	0	0.1
Water Reflections	2.1	0	3.2	0.2	72.2	0.1	0.1	1.1	3.6	12.9	0	0	4.6	0	0.1
Bark	0.9	0	0.3	0.7	81.3	0.9	0	0	1.6	5.3	0	0	8.1	0	0.7
Wood (cut)	0.6	0	6.6	0	65.7	0.5	1.5	7.9	1.9	4.2	0	0	10.6	0	0.4
Vehicle, Glass	3.1	0.2	0	0	24.9	2.7	0.1	0.4	0.3	0.1	0	0	67.9	0	0.4
Human	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Rock, Brick	1.3	0.3	0.2	0	12.8	0.4	0.7	2.5	0	0.5	0	0	9.7	0	71.6

Examining the F1 scores versus the class indices also affords insight to the behavior of the system. Figures 6.1.5 and 6.1.6 show the scores for the best and worst performing classifier respectively. It appears that the lower performing classifier loses the ability to predict specific classes: asphalt in shadow, bark, water reflections. While this is only a qualitative analysis at this point it could serve as motivation to collect more samples for these classes. This corroborates the majority classifier theory, as the smaller classes show a reduction in performance. However, the final convergence test should also be included here, and is in figure 6.1.7. It is not immediately clear from this graph if the smaller classes show a reduction in performance at the benefit of the larger. Most classes did drop in performance as well as the largest class. The variances of the F1 scores of the three experiments are $\sigma = 0.267$, $\sigma = 0.270$, $\sigma = 0.242$ respectively. While the changes in the variances are small, it is still clear that the largest class, Leaves, is one of the top performers. In addition, out of the total fourteen classes, five of the seven smallest classes perform the worst. The smallest 7 classes are Bark, Asphalt in Shadow, Grass in Shadow, Concrete, Water, Wood, Vehicles/Glass.

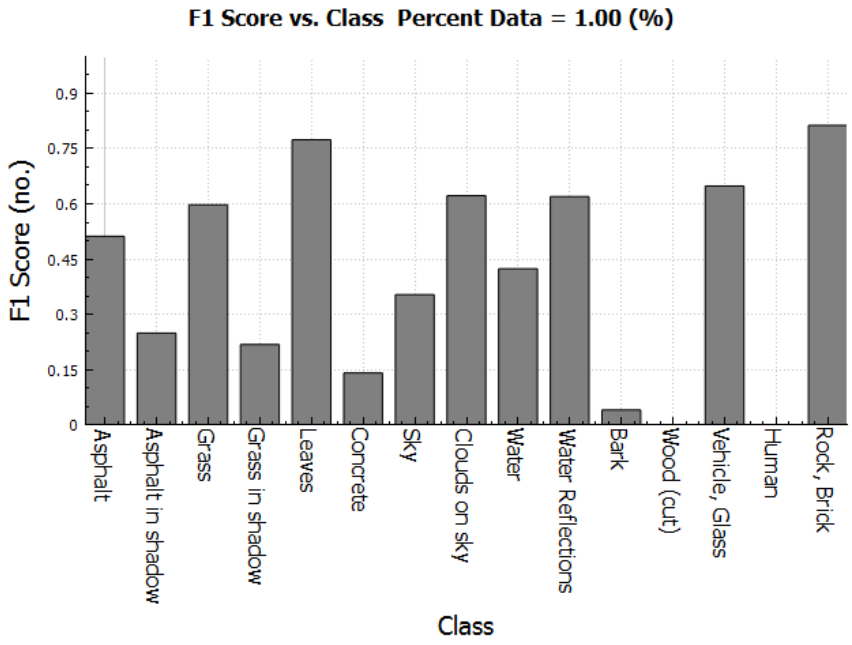


Figure 6.1.5: F1 score vs. class for data convergence, $data = 1\%$.

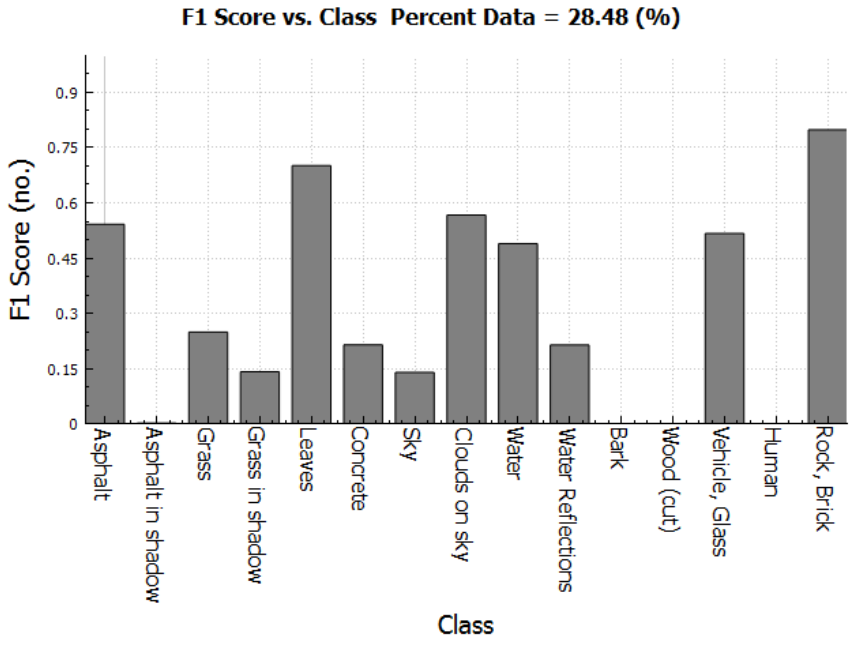


Figure 6.1.6: F1 score vs. class for data convergence, $data = 28.5\%$.

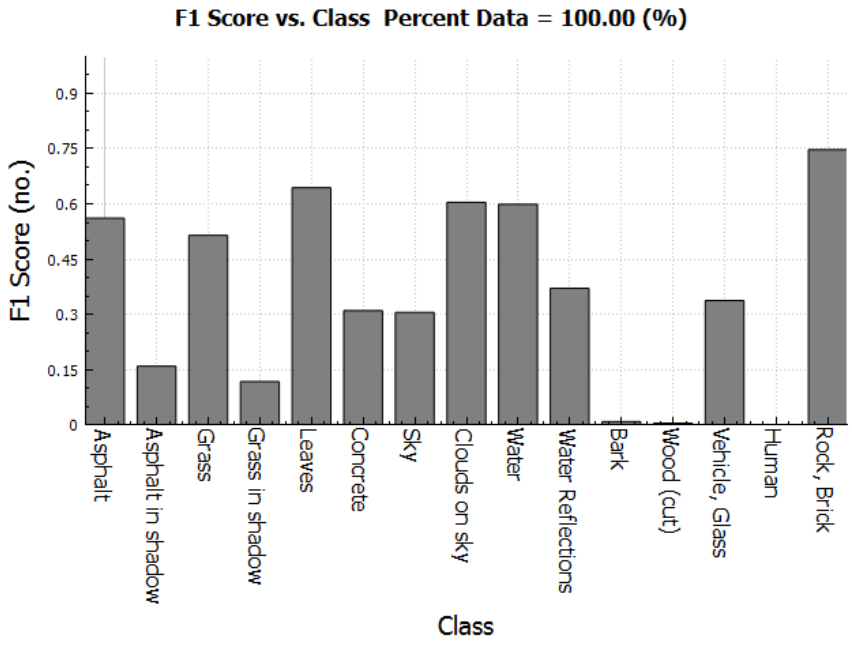


Figure 6.1.7: F1 score vs. class for data convergence, $data = 100\%$.

The cross validation results are shown in figures 6.1.8 and 6.1.9. The maximum fine cross validation score was 92.8% for the optimum performer (6.1.8) and 83.5% for the worst performer (6.1.9). The bounds for the coarse cross validation steps were not optimized and were consistent between experiments. The coarse bounds, the step value, and the number of iterations were chosen mainly arbitrarily but the coarse bounds were adjusted in the development stage to encompass the results with a noticeable border of low performance. This border is seen as the darker gray portions of the plots.

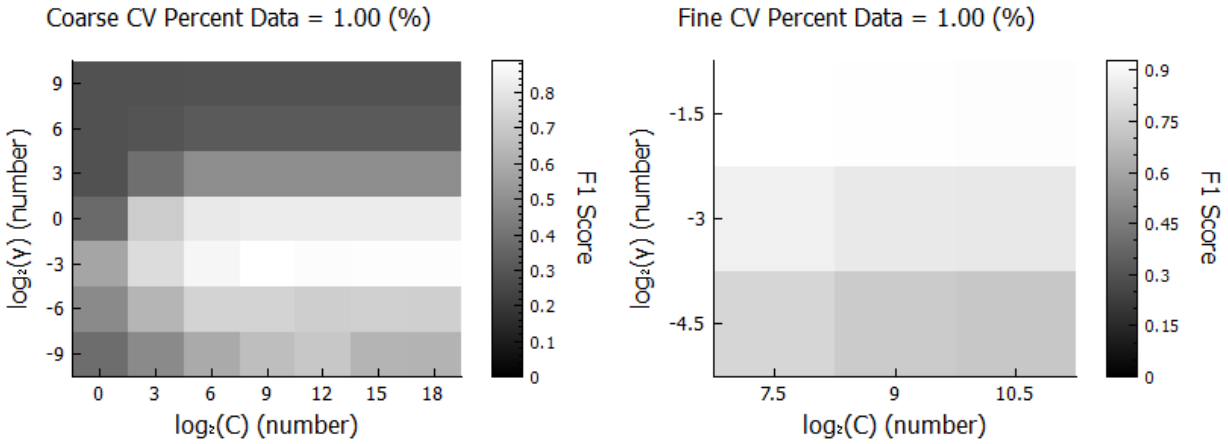


Figure 6.1.8: Cross validation for the data convergence experiment, $data = 1\%$.

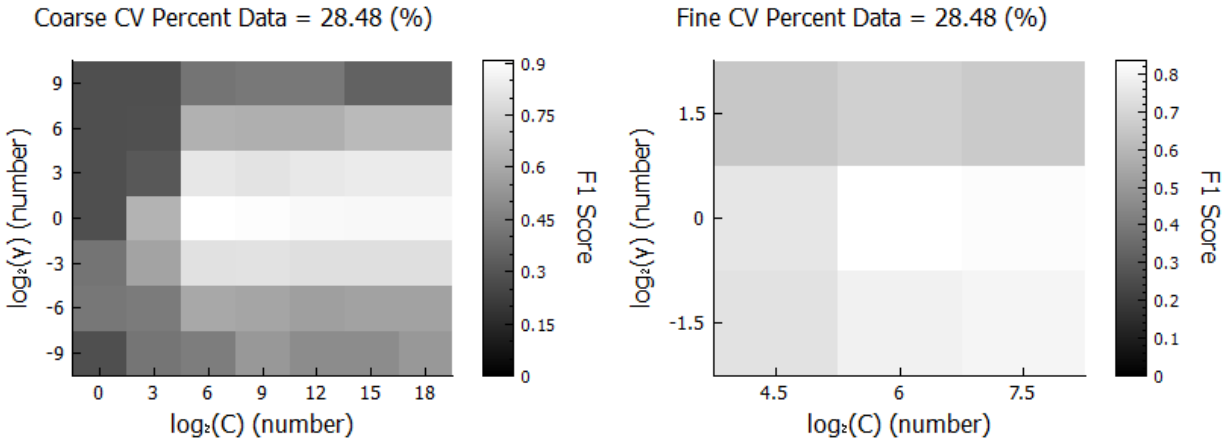


Figure 6.1.9: Cross validation for the data convergence experiment, $data = 28.5\%$.

The cross validation stages appear to have reached a local maxima for the γ value, but higher values of C appear to be suitable as well. A higher C value will penalize misclassification more. Higher C values would also penalize misclassification for the smaller classes higher than the larger classes due to the C-SVM class weights.

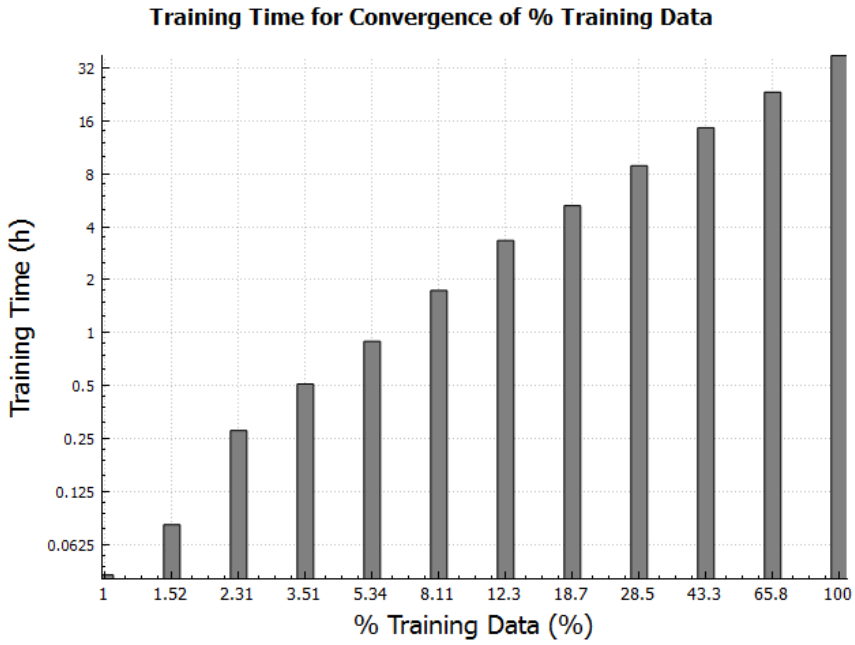


Figure 6.1.10: Training time vs percent training data, for percent training data convergence.

The cross validation stage should be considered for optimization as it represented the most significant time sink for the classifier training, meaning that the final training step took much less time than cross the validation step. The training time required for the complete system is plotted in figure 6.1.10. The smallest set took 2.5 *minutes* to train, the *data* = 28.5% set took 14.6 *hours*, and the *data* = 100% took 37.4 *hours*. The optimizations used in this study for cross validation were to under-sample the data for the coarse cross validation step and to multi thread all cross validation steps. The under-sampling was very effective given the preliminary tests, and the cross validation plots indicate that the local maximum was reached, albeit not with great precision due to the bandwidth. The multi threading may or may not have been effective but there was a more complete utilization of the cores.

It is useful to observe the result of the classifiers now that the relevant plots are shown. Sample predicted imagery is provided in figures 6.1.11 through 6.1.14 for the *data* = 1% classifier. Supporting images of the classifier inputs are shown alongside but not all the features can be shown here. DoLP and phase are included as they are often singular in the responses in that the feature segments the objects well and appear to influence detection strongly. Please note that the images are tiled, and the caption communicates the image order using MATLAB notation for matrices: commas for horizontal delimiters and semicolons for the next line.

Frame 6.1.11 is of a pathway with an overcast sky, grass and trees in the foreground and background. This is shown as the classifier does generally well segmenting the image, but there is significant confusion between the grass and tree canopies/bushes. Shadowed grass (gray-green) and grass (lime-green) is seen in the canopy, while the grass up on the hill is

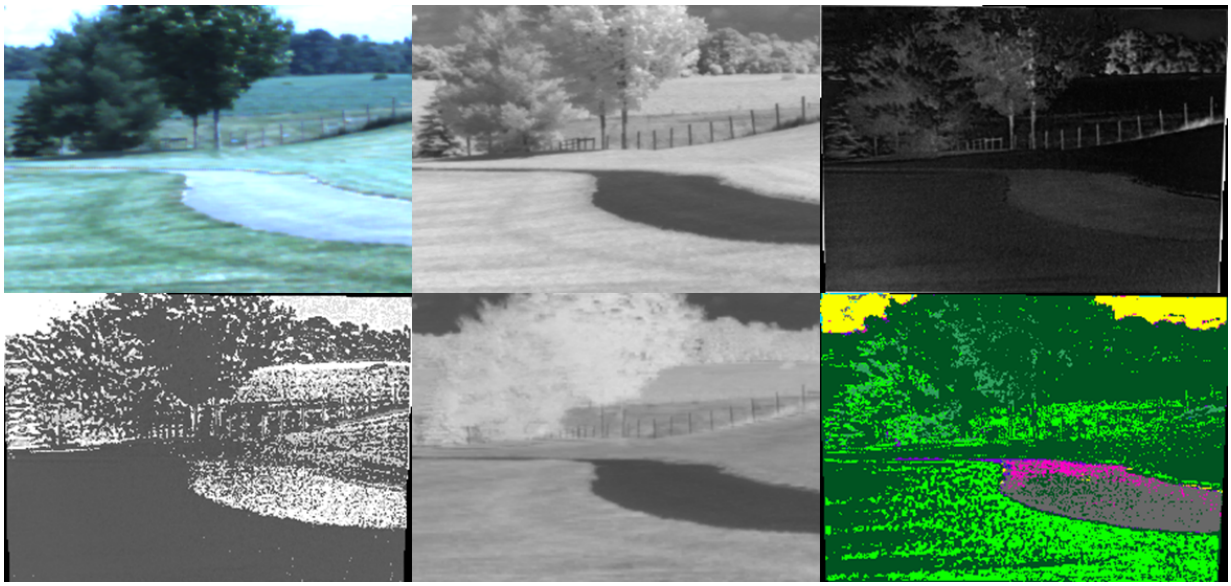


Figure 6.1.11: Sample results of the *data* = 1% classifier, pathway. Images are: [VIS, NIR, DoLP; phase, NDVI, prediction]

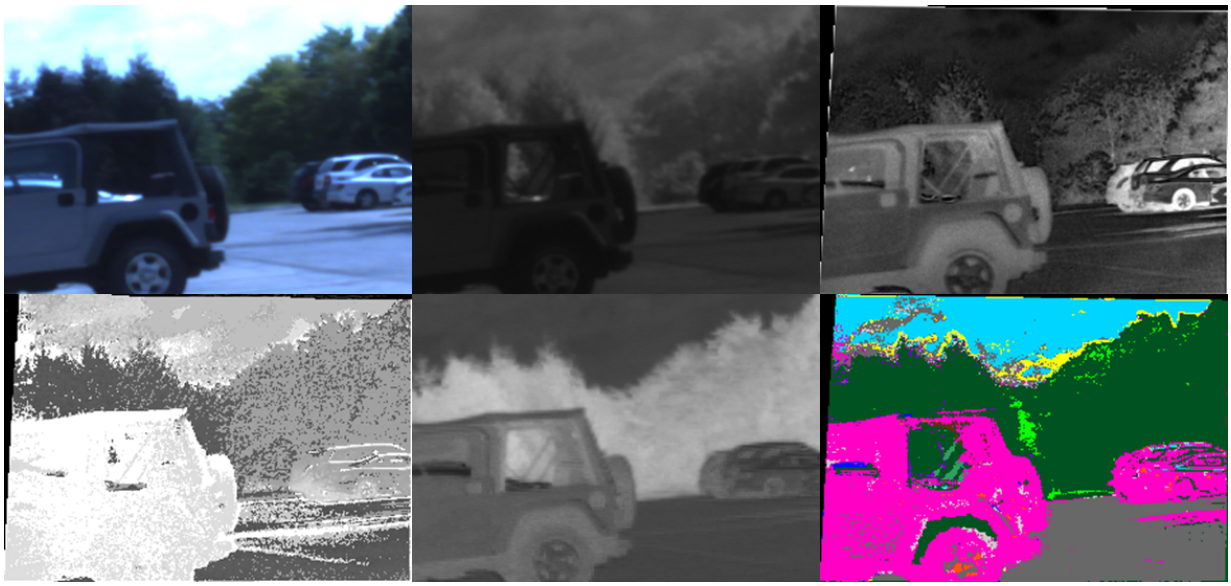


Figure 6.1.12: Sample results of the *data* = 1% classifier, parking lot with close up of cars. Images are: [VIS, NIR, DoLP; phase, NDVI, prediction]

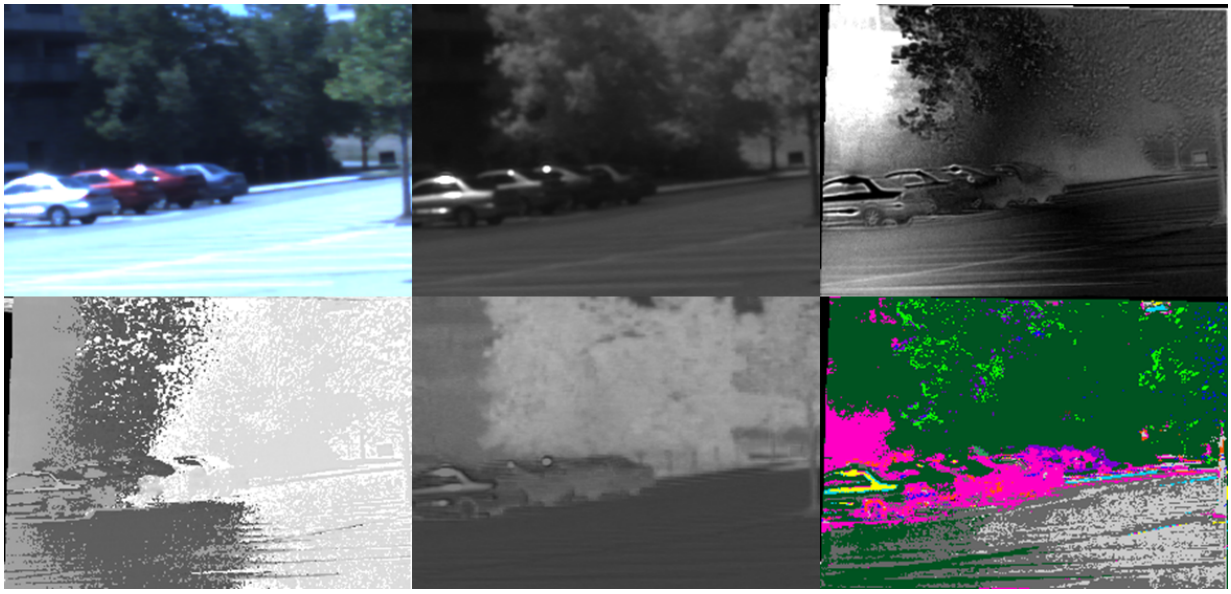


Figure 6.1.13: Sample results of the *data* = 1% classifier, parking lot with cars in the distance. Images are: [VIS, NIR, DoLP; phase, NDVI, prediction]

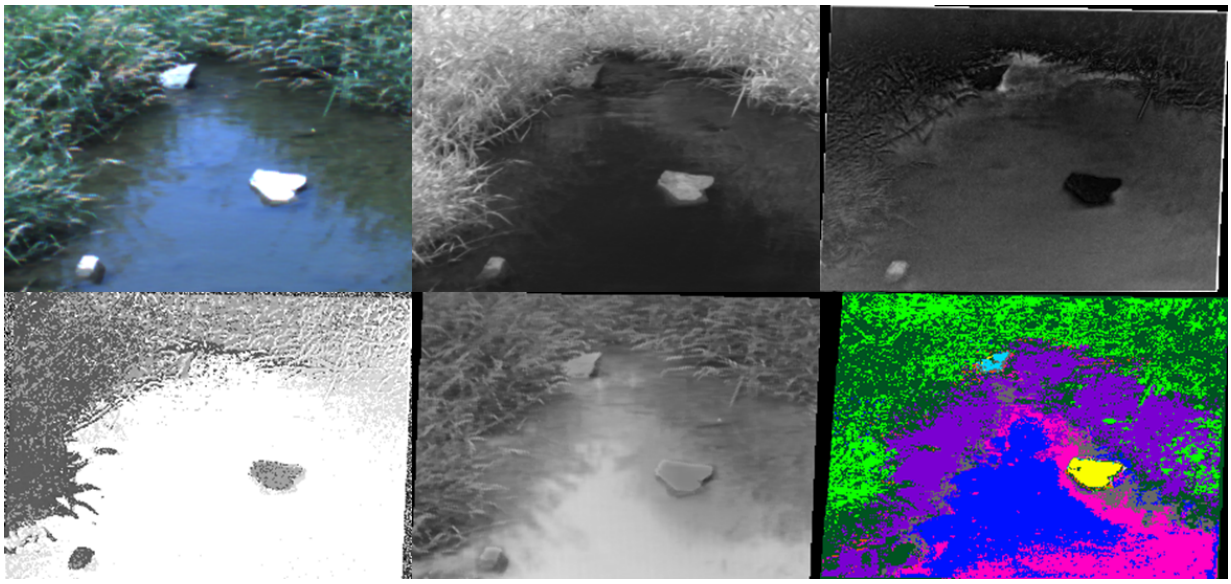


Figure 6.1.14: Sample results of the *data* = 1% classifier, reflections on water. Images are: [VIS, NIR, DoLP; phase, NDWI, prediction]

misclassified as a tree canopy. Note that the DoLP is very low in this area, which appears to affect segmentation. The pathway is predominately identified as asphalt but the small amount of magenta represents a vehicle.

Frame 6.1.12 shows vehicles near and far in a parking lot with nearly overexposed sky. The DoLP is bright especially for the glass and shadowed areas of the vehicles. The sky is classified mostly as clear blue with some asphalt in the top left corner. While the DoLP varies on the lot, it is classified well save for the shadow in the right. The wheel well of the near vehicle interestingly is misclassified as impassible vegetation.

Frame 6.1.13 is an example with more confusion, namely the asphalt/concrete in the foreground and the tree canopies (impassable vegetation). The building in the background that is strongly confounded by the shadow is also misclassified as tree canopies. The vehicles in the central part of the scene are mostly identified with the exception of the glass and the misclassification of the shadowed building as a vehicle.

Frame 6.1.14 is interesting in that many states of water and reflections are shown. NDWI is used in place of NDVI in this case. The main reflections are from the sky, but the reflection of the vegetation is unique as it is over such a shallow region and the bottom can be seen. The reflection is much stronger in the NIR however. The phase and NDWI both highlight the water remarkably well regardless of reflections. The predicted image shows reflections in the periphery as purple, water without reflections other than sky in blue, and then misclassified vehicles/glass as magenta. Vehicles and glass similarly reflect large amounts of light. Notice that the three rocks were misclassified each in three different ways: as blue sky, clouds, and trees. Rocks were included in the data set but the labels were predominately for brick, as bare rock such as these were not readily available.

6.1.1 Exploration on the Class Imbalance Problem

It is worth exploring the class imbalance problem further since the performance declined as the number of samples were increased and that the distribution of the samples per class biased certain classes. Furthermore many of the smaller classes were predicted as the larger class of index 4: leaves/impassible vegetation, suggesting that the classifier is exhibiting majority classifier characteristics. This issue is explored further by attempting to answer the following research questions: 1) Does a class imbalance exist? 2) Does the class imbalance problem impact the results? 3) Is it worth addressing the class imbalance? The first and second questions are explored through additional graphs showing the class ratios of the test set and the classification accuracy convergence. The last question is explored through a quick experiment with a training set that has been sub-sampled.

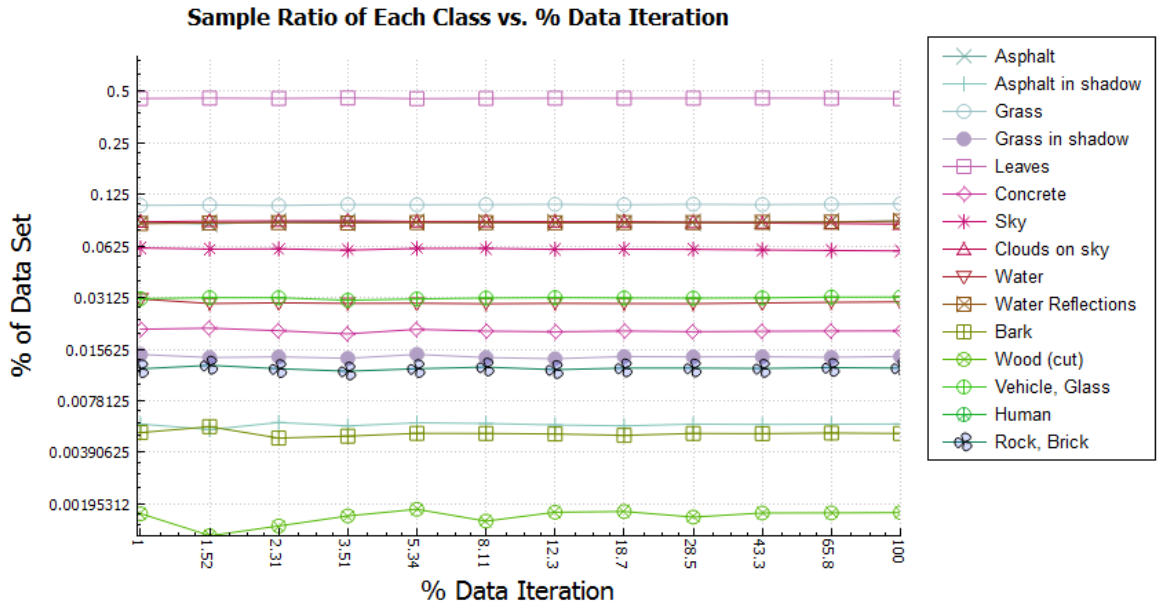


Figure 6.1.15: The percentages that each class is represented in the total training data set for each class vs. % data iteration.

Presence of a Majority Classifier

Firstly, what are the ratios of the classes and how does it change as the % data is converged? This is shown in figure 6.1.15 and shows that the ratios are stable. The human class again was not used here and the values are floating point errors, visible as the plot is log log. This graph shows that the largest class was consistently nearly 50% throughout the test. This suggests that any convergence is not due to the ratios changing and also suggests that indeed a class imbalance exists.

Secondly, how do the F1 scores for each class behave as the % data is converged? Do the scores of the larger classes drop faster than the smaller classes? The F1 scores should drop for both the larger and smaller classes. Recall that the F1 score of equation 3.45 should converge to 0 as confusion between classes increases; it is the harmonic mean of precision and recall. Contra to this is the convergence of classification accuracy which should converge to the percentage that the largest class is represented in the test set, at the most extreme result of the majority classifier. Figure 6.1.16 shows the F1 scores for every class as the training set increases. One can see that the smaller classes are sporadic while the largest class does decrease slightly. The smaller classes vary wildly, although all the F1 scores per class do decrease from 1% to 100% data with the exception of asphalt, concrete, water, and vehicles. It is not clear from this plot that the imbalance problem is affecting the classifier performance as the % data increases. This qualitative analysis merely corroborates the general trend seen in the confusion matrices.

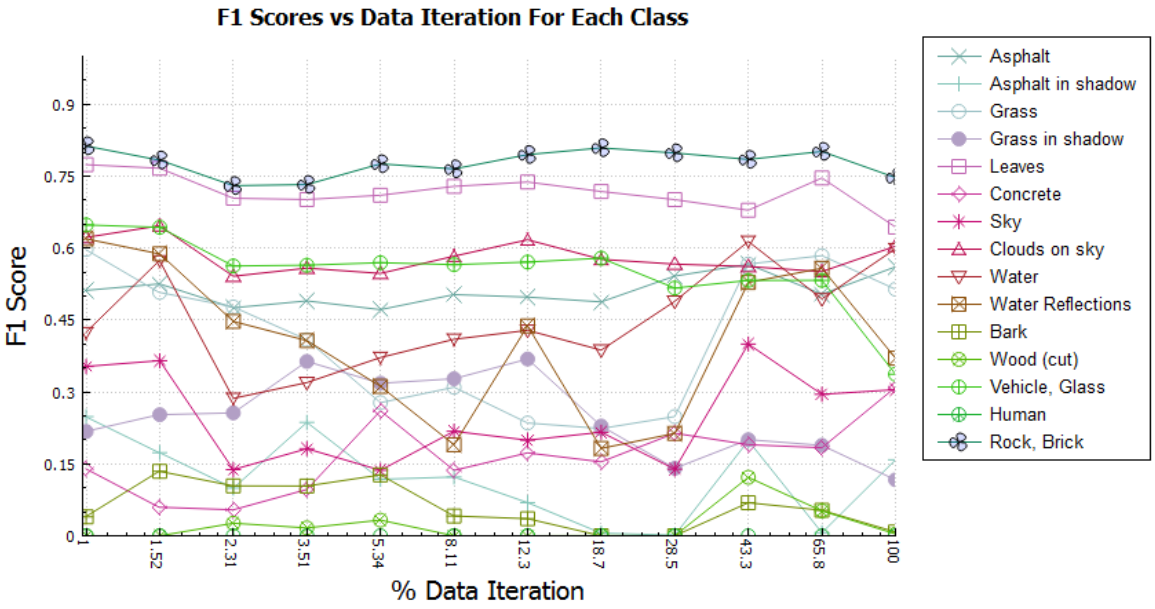


Figure 6.1.16: F1 scores for each class vs. % data iteration.

The classification rate however may be more informative as a majority classifier is predicted to converge to the percentage of the largest class in the test set. First, the ratios of the test set class representations can be seen in figure 6.1.17. The ratio of the largest class is 37.5%. Classification accuracy however is not defined on a per class basis: equation 3.41e shows that it is the sum of the TP and TN over the total sample count. The i^{th} entry of the diagonal will be the TP for class i and the remaining diagonal entries will be the TN samples. This results in equivalent classification accuracies if calculated on a per class basis as the numerator will be the sum of the diagonal entries. The classification accuracy is shown in figure 6.1.18 along with the F1-Weighted mean. The plot shows that the classification accuracy tracks the F1-Weighted Mean but is consistently lower than or equal to the F1-Weighted Mean. Recall that the latter penalizes FP and FN more strongly. The classification accuracy does indeed decrease but it does not converge to the percentage that the largest class is represented in the test set: 37.5%. This shows that the classifier performs better than a fully converged majority class classifier. This means that the most extreme majority classifier has not been reached but does not confirm or deny the presence of the imbalance problem.

Figure 6.1.19 shows precision and specificity as the training set size converges, see equations 3.41c and 3.41d. The two measures show the fraction of the predictions that were correct for those predicted as true and false respectively. This shows more detail than the classification accuracy as it can be computed on a per class level. It generally is similar to the classification accuracy plot however. One can see that the highest performing classifier is the Rock and Brick class, which also decreases slightly in performance as the training size is increased. The specificity of the entire classifier however has less variance than the precision. This indicates

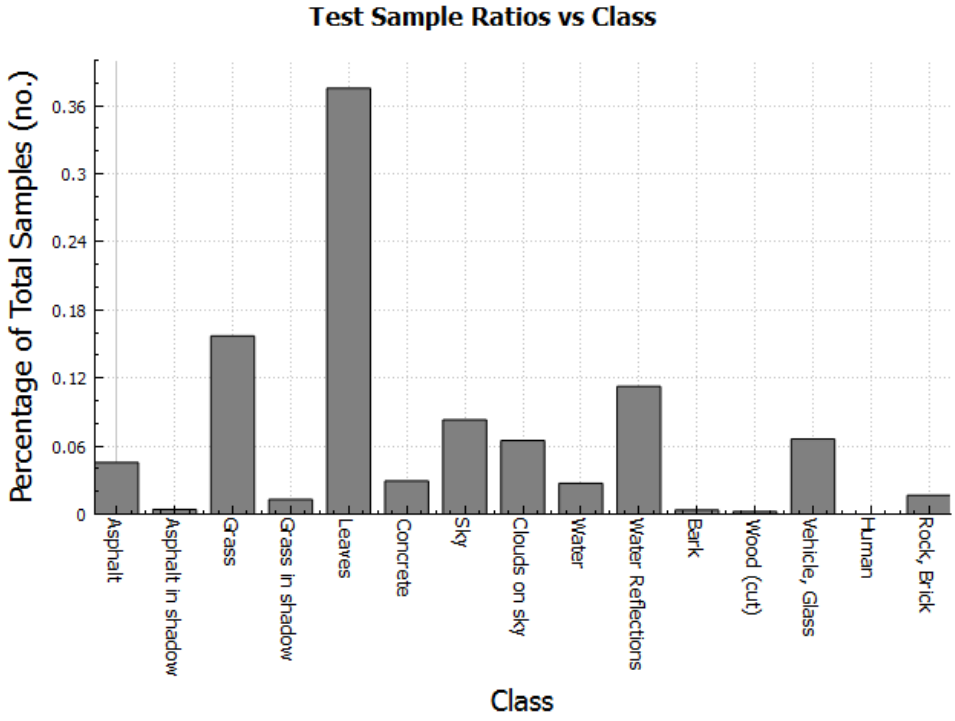


Figure 6.1.17: Ratios of the test samples per class for data convergence.

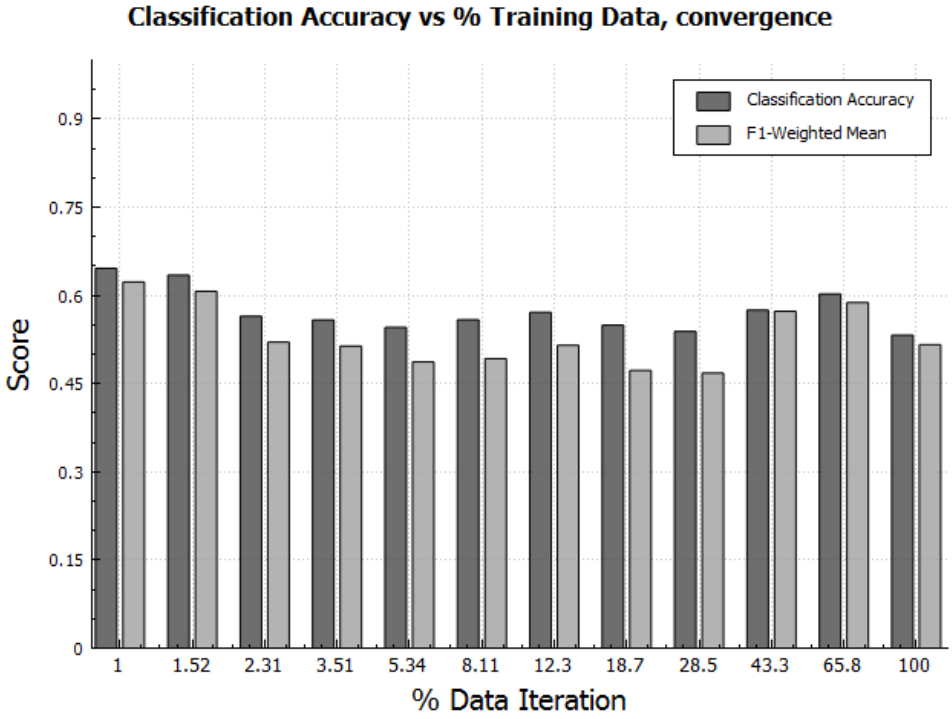


Figure 6.1.18: Classification accuracy vs. % data iteration.

Figure 6.1.19: Precision and Specificity vs. % data iteration.

that there exists a high variability that the true predictions are relevant, and also that the classifiers can more consistently identify negative samples between classes.

Qualitatively speaking there is not an obvious trend. However, because the sample ratios are consistent between the experiments it is not expected that the larger class should gain performance at the expense of the performance for the smaller classes, of which would occur if the class imbalance problem was indeed exacerbated. Therefore it is not clear from this initial analysis to what extent the class imbalance problem impacted the performance if at all. The leaves class, impassable vegetation, does indeed have a high performance but so does the rock/brick class. Smaller classes have low performance but a more in depth study of correlation has not been performed. To reiterate, the second research question being addressed is whether the class imbalance impacts results, but the larger question is whether it is worthwhile to explore this issue. The former would require a more in depth analysis looking perhaps at the correlation between sample ratios and performance and possibly oversampling one class to find a convergence similar to 6.1.4. Remember that in this accidental experiment the training set was oversampled due to a compounding effect where the training sample ratios did indeed change. However, here we are attempting to answer whether it is worth exploring the class imbalance problem and not to prove the existence of it. To that end, it can be said that the classifier has not converged to the worst case majority classifier scenario as the classification accuracy did not converge to the percentage

that the largest class was represented by in the test set. Because it can be said that such a class imbalance could exist upon inspection of the training sample ratios, it is sufficient to proceed to another exploratory test where training sample imbalance is reduced.

Exploratory Balanced Set

The third research question in this section is whether it is beneficial to pursue a balanced training set. To this end the training set was sub-sampled in order to discard samples above the median frequency. This exploratory test was completed for 1%, 5%, and 100% of the training set, although this does not mean that the percentages are comparable between this exploratory balanced test and the previous % data convergence. The number of samples used in this exploratory balance convergence corresponds to 0.293%, 1.45%, and 30.9% of the original training set respectively. The last iteration shows that 69.1% of the training samples were thrown out in order to clip any samples above the median value. The test set was not sub-sampled and was the same as the all the tests in this study.

The number of training samples for the largest sub-sampled training set is shown in figure 6.1.20 and the ratios of those with respect to the total training set is in figure 6.1.21. The number of samples vs. class plot shows that 6 classes were clipped at the median value but the 7 are still significantly smaller in number, and the remaining class, Water, was nearly equal to the median. The exploratory performance convergence plot for this is shown in figure 6.1.22. The plot shows that the performance does not appreciably change, and is approximately $F1_W = 0.462$ and $CA = 0.424$. This is less than the performance of the previous classifier at the nearest number of samples 28.5%, $F1_W = 0.538$ and $CA = 0.467$. The standard exponential convergence does not occur in this exploratory experiment. Instead the performance is essentially constant, and lower than without sub-sampling, which suggests that the balancing may not be beneficial. However, there are other alternatives.

First, the balancing is incomplete. While the variance of the sample counts did indeed decrease, half the classes did not reach the median value.

Second, the test set and the training set now are from different populations, meaning the training set does not model the test set well. Modification of the test set was not explored in order to maintain consistency. However, the primary task with the test/train split is to obtain two samples from the same population. A reduction in performance would be expected if this is not acheived.

Third, the samples were split on a per frame basis but converged on a percent of each frame basis. This was done early in the programming stage as each class is not represented equally or at all in each frame. Furthermore, the data could not be split on a per pixel basis as the pixels from one frame are highly correlated. It is very likely that two pixels in close proximity will either be of the same class or provide information on the other. Using pixels from the same frame would be essentially testing on the training data, of which early experiments had

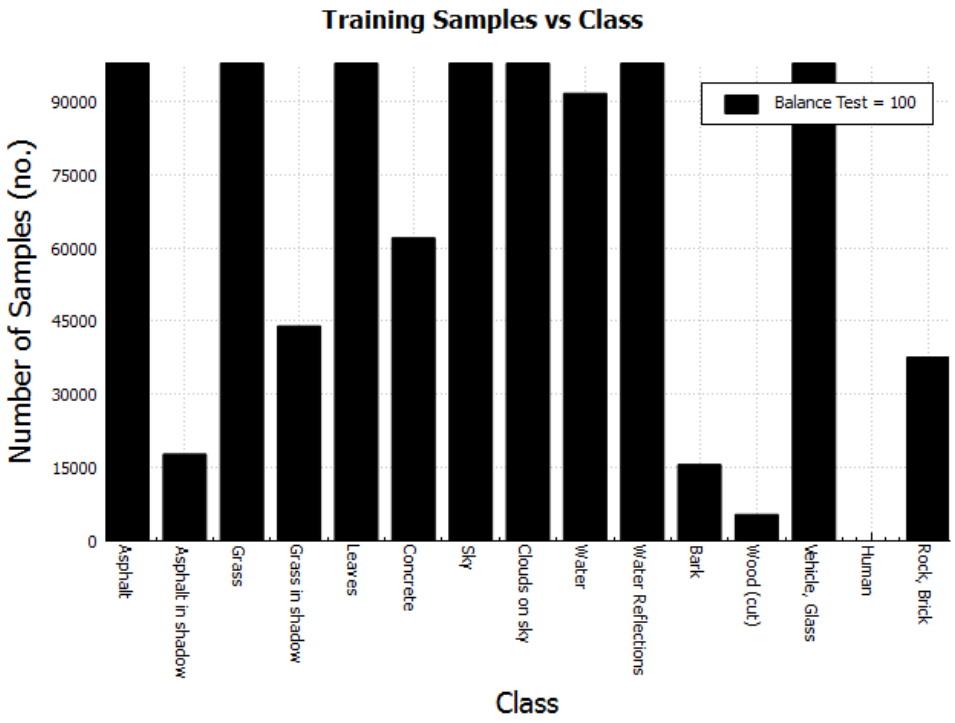


Figure 6.1.20: The number of training samples per class for the largest balanced set.

shown. Remember that the % data was converged on a per frame basis, and the exploratory balanced test in 6.1.22 illustrates that the performance does not increase as the number of pixels from each frame increases. This means only a small number of pixels from each frame are required to reach a certain performance and higher percentages merely increase training time. This makes sense intuitively if one notes again that the pixels from one frame are correlated with one another. Oversampling the same pixels will not necessarily improve the classifier.

Finally, the analysis could be wrong and there is another explanation. This does not mean that it is indeed this case but it is always important to keep it in mind.

6.1.2 Conclusions

The primary issue is that the expected convergence in performance was not observed as the training set size increased. The class imbalance problem was explored and it was found that only a small percentage of each frame is likely to be required in order to reach a certain performance, seen in figure 6.1.22. It is possible that the reduction in performance between the exploratory balanced iteration and the previous unbalanced iterations is due to the large difference in the distribution of the testing and training sets. The balanced set distribution changed drastically while the test set was constant. However, it is more important to note

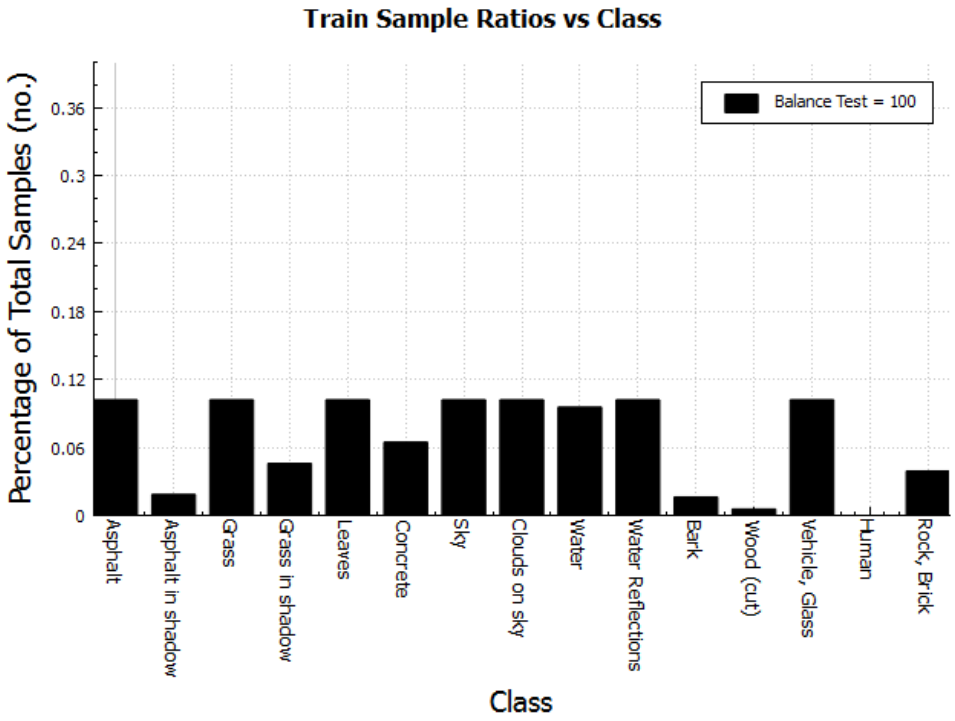


Figure 6.1.21: The training sample ratio for the largest balanced set.

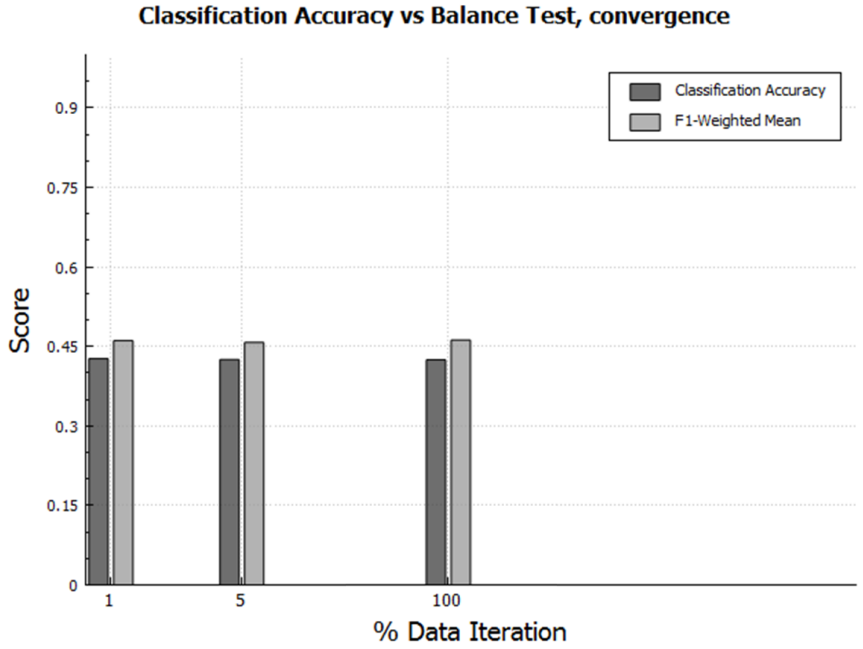


Figure 6.1.22: Classification accuracy and F1-Weighted Mean vs. % balance iteration.

that the methods for splitting data into testing/training sets and the method of including more data is likely flawed. As stated in 6.1.1 Exploratory Balanced Set: this study split on a per frame basis and converged on a percentage of each frame. This was done early in the study as not every frame exhibits every class equally, and yet small amounts of training/testing data was needed to develop with. An alternative of splitting each frame into test/train pixels would not be acceptable as pixels within a frame are highly correlated with one another, i.e. it would be testing on training data. A more complicated sampling system will be required in order to both combat the class imbalance problem and to produce a representative data set of all the classes.

Future work should consider changes to the test/train ratio, improvements in the labeling process, the class imbalance problem, and cross validation on the entire system. The test set used 40% of the total data available, compared to the recommended 20% by [100] and also used by [105]. A more complicated test set can reduce perceived performance but better approximate out of sample performance. Some of the confusion between classes could have been avoided by modifying the classes used, which is discussed further in 7.1.1 Class Selection. There is enough evidence to pursue the avoidance of the class imbalance problem further. This could be addressed via data collection and in the sub sampling process, discussed further in 7.1.3 Balancing & Sub-Sampling. Finally, cross validation on the entire system could be pursued to reduce the effects of data partitioning on the performance metrics. It has been shown here that the data partitioning method can be very influential on the results, and this would be an avenue for reducing such affects. This is discussed further in 7.1.4 Cross Validation for the Entire Data Set.

The results show promise in the classification and segmentation capabilities of the given targets despite the peculiar learning convergence. Qualitatively speaking, it appears that the images can be segmented well and a significant amount of confusion results from similar classes such as clear sky/couds and forms of vegetation. There are ample avenues for improvement. Some are fundamental such as the sub-sampling methods, and cross validation on the entire data set. Others are straightforward, tailoring data collection and the percentage train/test for example. It would be interesting to see such avenues come to fruition.

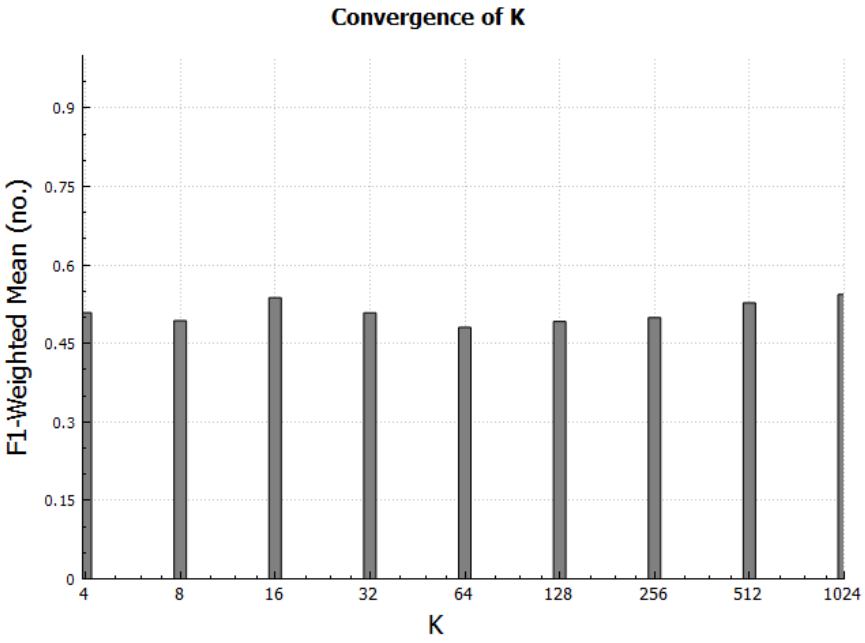


Figure 6.2.1: Convergence plot of K texton centers against F1-Weighted Mean for the classifier.

6.2 K Convergence

The number of textons K was swept logarithmically on $[4, 8, ..., 1024]$ while the percent data used was maintained at 10%. The analysis begins with the convergence plot of F1-Weighted Mean vs K, figure 6.2.1. The performance does not converge as it had for Shotton, Winn, Rother and Criminisi [14] where an initial increase in performance was followed by a peak, small loss in performance and a settling period. Instead the performance hovers around 0.50. This is particularly unusual as 4 textons have generally the same performance as 512. Despite the small variation in performance the maximum score was achieved at $K = 1024$ with $F1_W = 0.543$. The next highest performer was $K = 16$ with $F1_W = 0.537$. The lowest performer was $K = 64$ with $F1_W = 0.481$.

The test set is the same as the percent data convergence, and so is not reproduced here. The training set used was all registered pixels from the training set, and so was also constant. This is provided in figure 6.2.2.

The confusion matrices and the F1 scores vs class can show how the selection of K effected intraclass confusion. One can see from the F1 plots that there is not a stark difference in the scores with respect to the classes between the different K values, but it does vary.

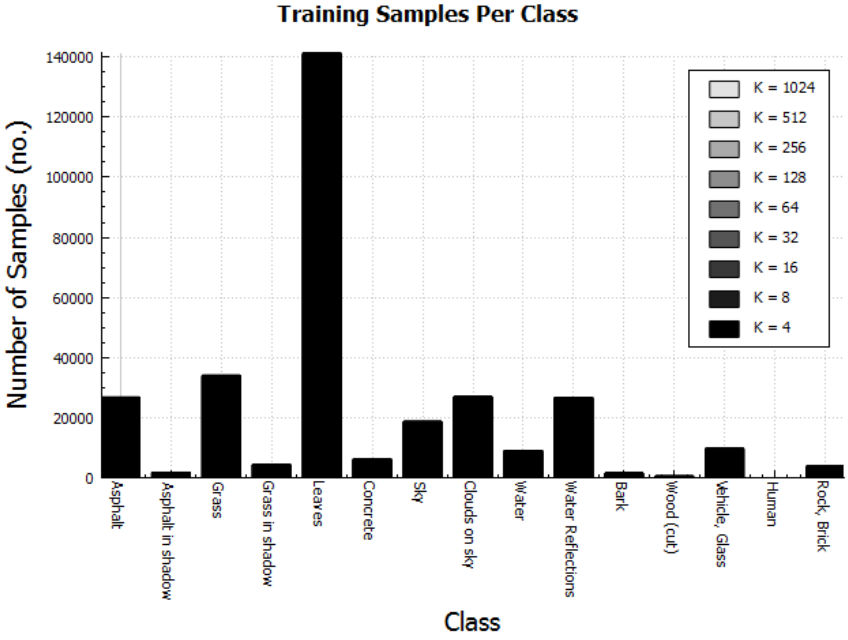


Figure 6.2.2: Convergence plot of K texton centers against the number of samples per class.

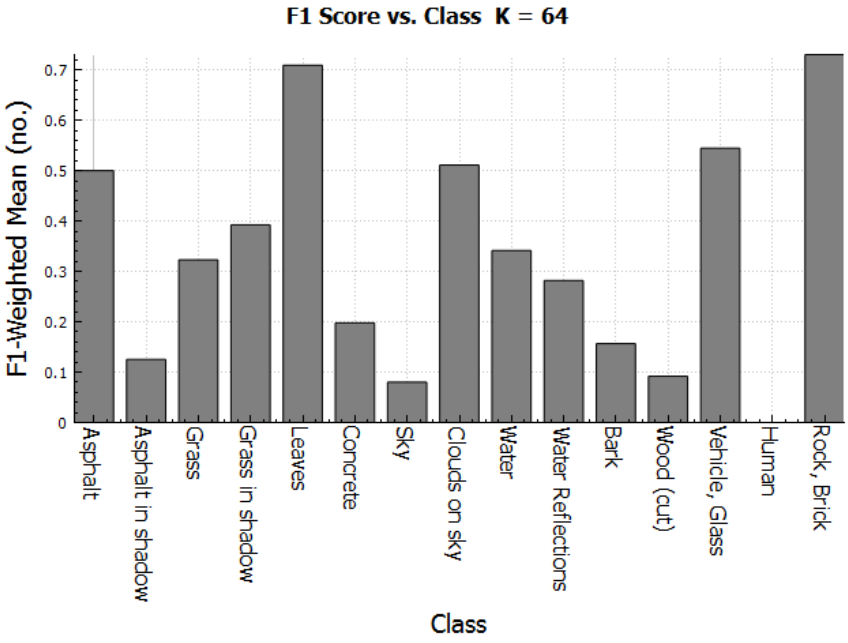


Figure 6.2.3: F1 score vs class index for the K convergence, $K = 64$.

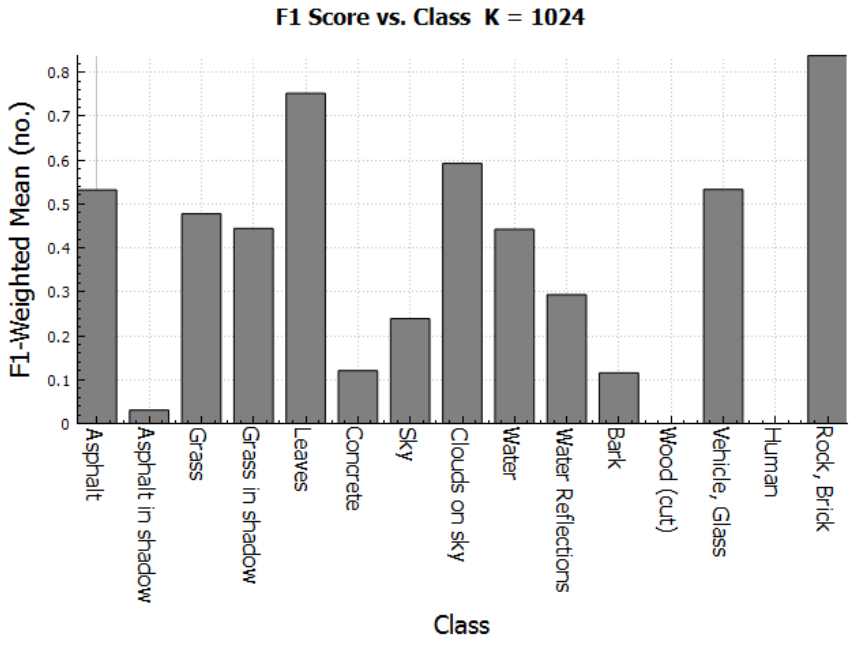


Figure 6.2.4: F1 score vs class index for the K convergence, $K = 1024$.

Table 6.3: Confusion matrix for K convergence, $K = 64$. Row normalized by the number of positive samples, reported as percent (%). Rows: truth, Columns: predicted.

	Asphalt	Asphalt in shadow	Grass	Grass in shadow	Leaves	Concrete	Sky	Clouds on sky	Water	Water Reflections	Bark	Wood (cut)	Vehicle, Glass	Human	Rock, Brick
Asphalt	64.1	0	0	0	18.1	5.9	0	0	0	0	0	0	11.2	0	0.6
Asphalt in shadow	2.1	11.6	0	0.1	16.3	2	0	0	0.1	0.1	0	0	67	0	0.7
Grass	0	0	22.7	0.5	75	0	0	0.1	0.3	1	0.1	0.1	0.2	0	0
Grass in shadow	0	0	2.8	33.2	55.5	0	0	0	0	8.1	0.1	0.3	0	0	0
Leaves	0	0	6.4	0.5	91.2	0	0	0.1	0.1	0.8	0.6	0	0.3	0	0.1
Concrete	52.5	0.6	0.1	0.1	12.2	15	0.9	1.9	0.1	0.3	0.2	0	15	0	1.3
Sky	21.5	0	0.7	0	23.1	3	4.7	20.8	0	0	0	0	26	0	0
Clouds on sky	9.2	0	0	0	9.3	4.6	17.1	44.5	0.7	0	0	0	14.5	0	0
Water	1.9	0.2	0.1	0.3	17.1	0	0	0	22.4	1.7	2	0.9	53.5	0	0
Water Reflections	0.8	0	2.4	1.5	60.8	0.5	0	0.2	1	17.4	3.2	0.3	11.7	0	0.1
Bark	1.4	0	0.7	3.7	56.6	0.1	0	0	0.6	3.6	24.2	0	8.8	0	0.3
Wood (cut)	0.9	0	3.5	1.2	60	1.7	0.2	2.7	0.2	1.7	2.3	7.2	18.2	0	0.2
Vehicle, Glass	1.4	4	0	0	10.2	2.6	0.1	0.3	0.1	0.1	0	0	80.6	0	0.5
Human	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Rock, Brick	1.4	0.4	0.2	0	9.9	0.2	0.1	3	0	0.6	0	0.5	20.7	0	62.9

Table 6.4: Confusion matrix for K convergence, $K = 1024$. Row normalized by the number of positive samples, reported as percent (%). Rows: truth, Columns: predicted.

	Asphalt	Asphalt in shadow	Grass	Grass in shadow	Leaves	Concrete	Sky	Clouds on sky	Water	Water Reflections	Bark	Wood (cut)	Vehicle, Glass	Human	Rock, Brick
Asphalt	69.2	0	0	0	1.7	2.8	0	0.1	0.1	0	0	0	25.8	0	0.3
Asphalt in shadow	2	1.8	0.1	0	2	2.3	0	0	0.1	0.1	0	0	90.9	0	0.7
Grass	0	0	36	0.2	58.9	0	0	0.3	0	1.3	0	0	3.3	0	0
Grass in shadow	0	0	1.2	31.2	50.8	0	0	0	0	16.8	0	0	0	0	0.1
Leaves	0	0	5.6	0.1	90	0	0	0.1	0	2.7	0	0	1.2	0	0.1
Concrete	56.4	0.1	0.1	0	4.2	7.5	4.8	2.3	0.2	0.6	0	0	23.2	0	0.6
Sky	20.1	0	0	0	1.1	0.2	16.1	21.1	0	0.5	0	0	41	0	0
Clouds on sky	7.7	0	0	0	4.4	2	21.1	55.3	0.3	0.1	0	0	9.1	0	0
Water	1.2	0	0	0	11.9	0	0	0	33.3	4.2	0.8	0	48.5	0	0
Water Reflections	0.8	0	1.1	0.3	63.9	0.1	0	0.5	3.6	19.8	0.4	0	9.4	0	0
Bark	0.5	0	0.4	0.4	65.7	0.5	0	0	0.3	14.3	7.5	0	10	0	0.3
Wood (cut)	0.8	0	13.1	0	43.3	0.4	0.8	5.8	1	4.8	0.1	0	29.4	0	0.5
Vehicle, Glass	2.5	0.8	0.1	0	1.6	3.4	0.2	0.3	0.1	0.1	0	0	90.5	0	0.5
Human	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Rock, Brick	0.9	0.3	0.2	0	5.5	0	0.5	3	0	0.9	0	0	12.5	0	76.2

The confusion matrices are similar to the percent data convergence and are not remarkable here. There is a large amount of confusion as always with the leaves, a.k.a. impassable vegetation, but statistics will likely be required to make any finer conclusions given the data. If the differences were greater or a more in depth analysis was done, one might expect to see which classes are affected by texture. Certain classes like vegetation and reflections are good candidates. Grass and tree canopies have similar colors but different textures; some texture is lost upon reflection.

The training time for the textons is on the order of minutes and did not vary significantly contrary to the cross validation step in the convergence of percent training data. The training time for the complete system did vary however and is shown in figure 6.2.5.

The cross validation results are shown in 6.2.6 and 6.2.7. The maximum $F1_W$ score was $F1_W = 0.869$ for $K = 64$ and $F1_W = 0.873$ for $K = 1024$. The plots appear similar to the percent data convergence in that the γ value has likely reach a maximum, but higher values of C may be appropriate. This is seen by the large peak extending from $\log_2 C = 6, \log_2 \gamma = 0$ to larger values of C.

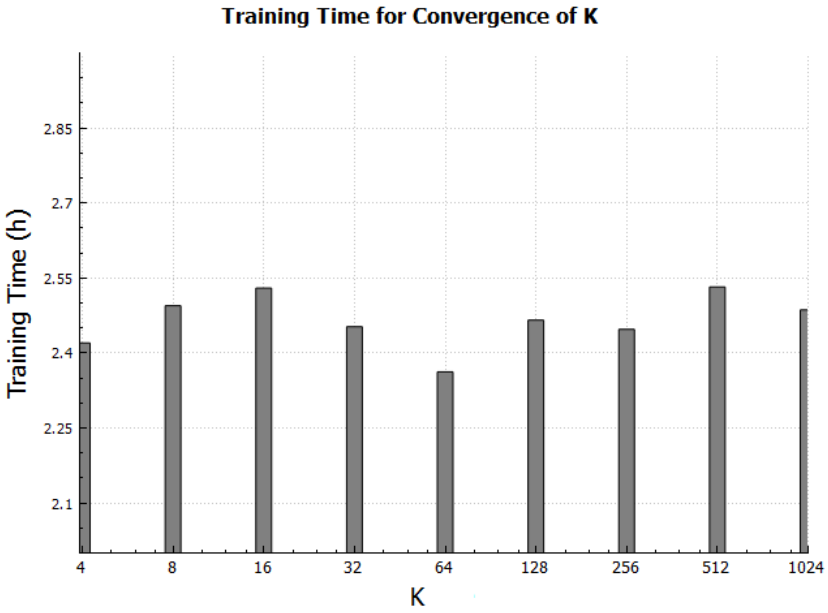


Figure 6.2.5: Training time vs K texton centers for K convergence.

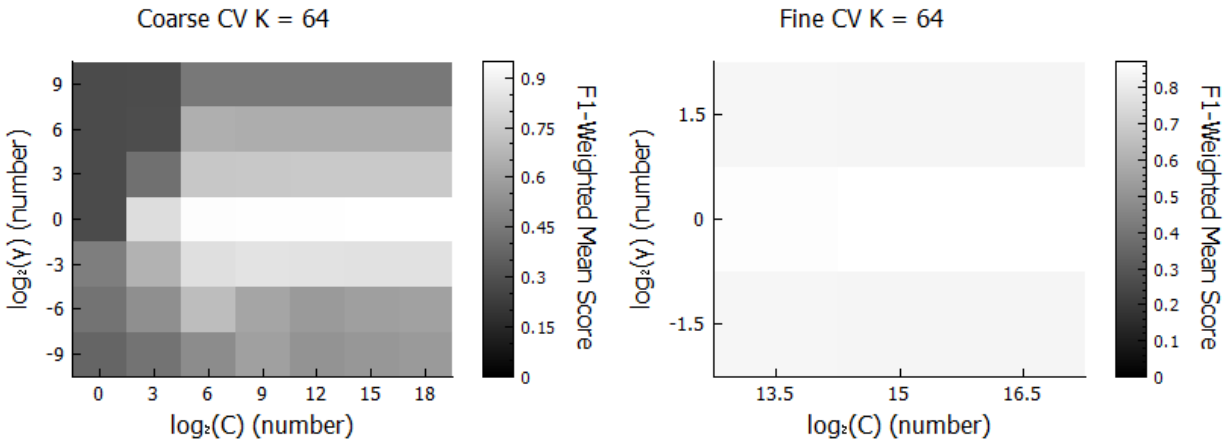


Figure 6.2.6: Cross validation for the data convergence experiment, $K = 64$, $data = 10\%$

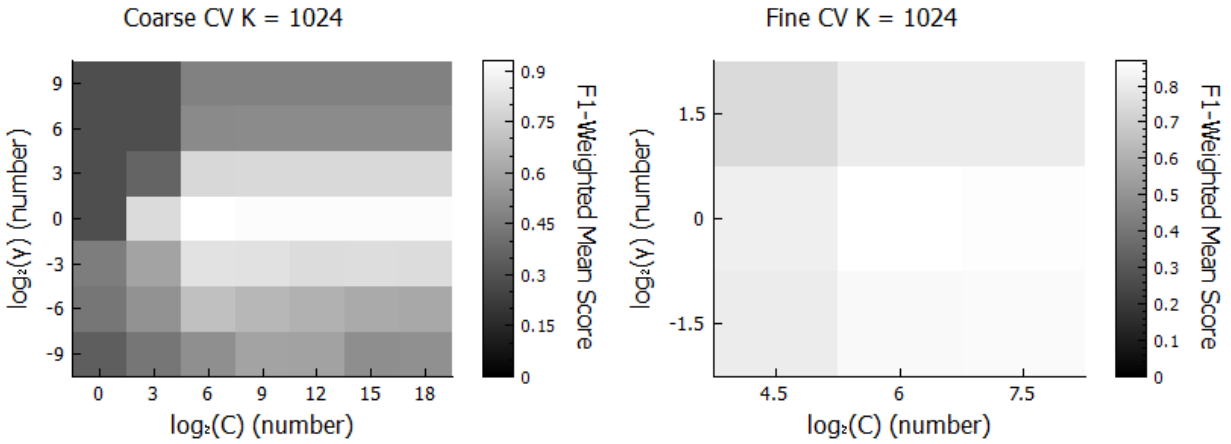


Figure 6.2.7: Cross validation for the data convergence experiment, $K = 1024$, $data = 10\%$

Analysis

The convergence of the K number of textons did produce similar behavior of that in Shotton, Winn, Rother and Criminisi [14] for a small scale of $K = []4...64]$. Performance peaked in Shotton et. al. initially after the first increase of K and then tapered off as K was increased where it was concluded that "with too many textons the [classifier] starts to overfit"[14]. The performance here however exhibited an upward trend once more as K was increased from 64 to 1024, beyond the bounds found in Shotton et. al.: $[25, 50, 100, 200, 300, 400]$.

There are two possibilities discussed here that could either identify a new course of action or explain this behavior: 1) The categorical data was not encoded properly 2) The texton information segments well but does not classify well.

This behavior of the K convergence is significant given the recommendation concerning categorical data of Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin of LIBSVM[97]. Chih-Wei et. al. noted that SVMs require real numbers of which categorical features must be converted to. Rather than using integer values, they "...recommend using m numbers to represent an m-category attribute...where one of the m numbers is one, and other are zero [as it] might be more stable than using a single number"[97]. This study encoded the texton value in a single integer. The vector could be realized however if the value of the vector was 1 at the index equal to the texton and zero else. This encoding strategy could impact the results but might not be suitable for the large values of K used here. Comparison of the convergence to Shotton et. al. would benefit from testing the vector encoding as well as the integer encoding.

The texton information may be enabling high segmentation performance but not classification, meaning that the scene is partitioned well in local areas but those areas are not identified as the correct class. This can be seen qualitatively in figure 6.2.8. A larger quan-

titative analysis would require a prohibitively detailed ground truth for the purposes of this study. Nonetheless, the trend seen in this figure is indicative of the larger trend observed: the number of segments in the prediction image increase as K increases. Figure 6.2.8 shows a test frame of the K convergence where a false color of the texton plane is shown alongside the prediction false color, $K \in [4, 16, 64]$. $K = 4$ shows all the vegetation predicted as leaves, the asphalt predicted as a vehicle, and the sky somewhat variegated. As K is increased the grass regions begin to be misclassified as something other than leaves and the clouded regions are eventually are classified more accurately. Qualitatively speaking, this effect varies wildly: one class may lose accuracy while another gains as K changes. The power over the textons to segment well makes sense as K-Means is often used for super pixel clustering. This segmentation may not be reflected in the classification accuracy/ F1 Scores due to the high rate of confusion. This shows that optimization of this hyperparameter K is likely to be significant in future work.

6.2.1 Conclusions

The over-fitting trend seen in Shotton, Winn, Rother and Criminisi[14] was not seen for the K convergence test. A peak was found at $K = 16$ but performance increased again at $K > 64$. Future work may benefit from encoding the categorical data in a binary vector and considering the possibility that the texton information may aid segmentation more strongly than classification. Future work should consider running the learning convergence with and without the texton information in order to explore the usefulness of this feature.

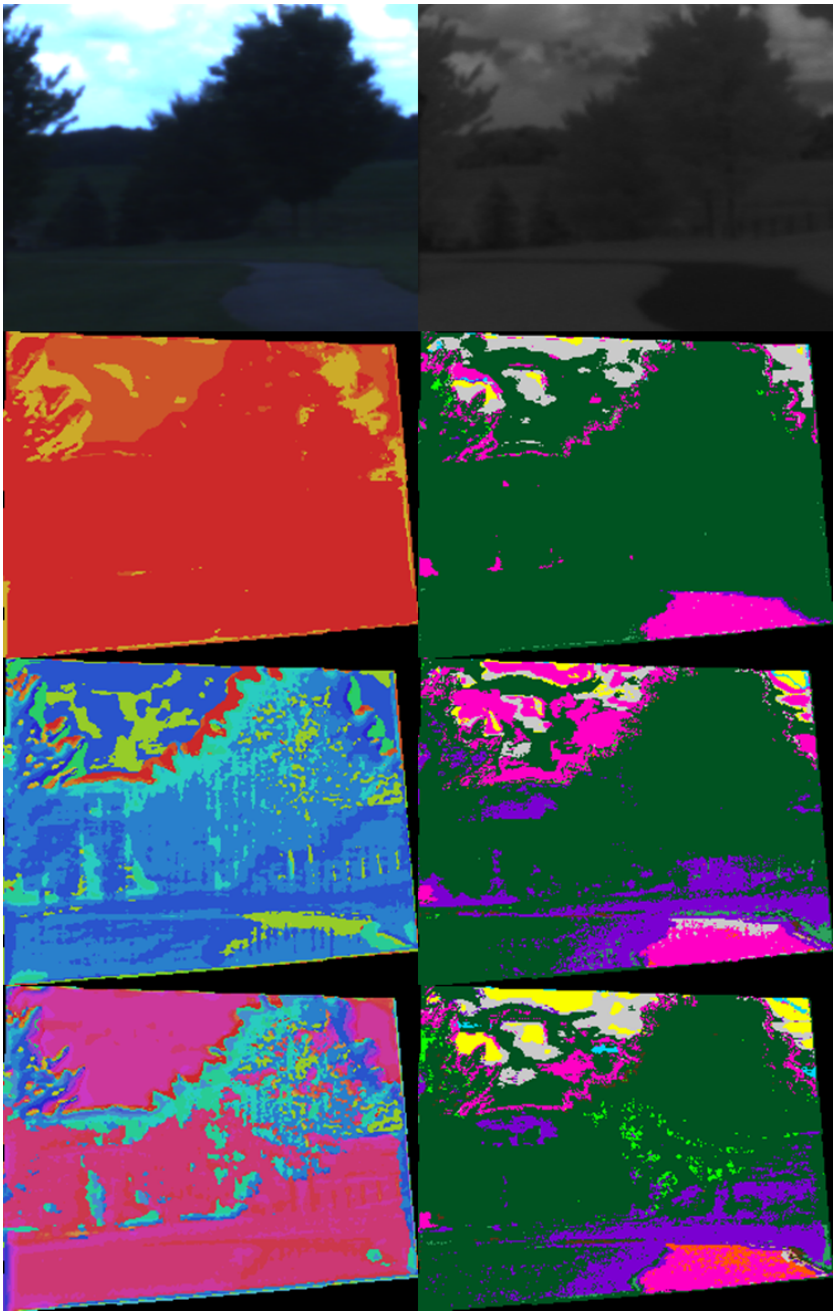


Figure 6.2.8: Example texton and prediction planes. Results shown for $K \in [4, 16, 64]$ from the K convergence test. The images are: [VIS, NIR; Texton $K = 4$, Prediction $K = 4$; Texton $K = 16$, Prediction $K = 16$; Texton $K = 64$, Prediction $K = 64$]

6.3 Closing Remarks

A complete system has been demonstrated from data collection to analysis of performance. Multispectral and textural features were extracted and trained on an SVM. The original goal of producing a complete system for pixel level classification in UGV settings has been achieved. Hyperparameter selection was performed through the texton convergence and learning convergence, in addition to cross validation for SVM specific hyperparameters. Many methods for evaluating and inspecting performance were produced. The expected behaviors for texton convergence and learning convergence were not observed, but important conclusions came from both. For texton convergence, the hyperparameter K did not necessarily exhibit over-fitting as K increased but did influence the performance. The encoding strategy could be modified and segmentation performance metrics could be pursued. Regardless, the feature appears to influence segmentation strongly. The learning convergence was explored in depth and concluded that a more complicated sub-sampling method for gathering training/testing samples should be pursued in order to combat the balancing problem and more importantly a proper convergence on the number of samples. The exploratory balanced set strongly implies that only a small subset of pixels from each frame will be required to reach a given performance level. If this is so then it will be very beneficial for reducing training time. There are many other avenues for improvement, such as tailoring data collection to the class distribution of the data set, that are discussed further below. Overall the test imagery appeared very promising, often producing predictions that not only segmented the scene well but identified vegetation, vehicles and the overall type of material well. It would be very interesting to see the potential improvements pursued as well as applied to a system in the field.

Chapter 7

Future Work

7.1 Approach for Future Learning Convergence

As discussed in 6.1, there are many possibilities that could explain the disagreement of the learning convergence with the literature. Key areas are addressed here in order to explain the recommended approach for improving this work.

7.1.1 Class Selection

Firstly, there was significant intra class confusion between similar classes. The targets were similar in both type and response. There were three forms of vegetation for example, which was also confounded by shadows. Asphalt and concrete are similar in color and texture and are susceptible to overexposure, the same goes for the sky and clouds. These two pairs as well as the four classes were confused between one another. The examples shown here however do not necessarily have to be separated. What is the functional capability gained from differentiating the sky and clouds? Asphalt and concrete? Vegetation in and out of shadow? In addition, some classes were very difficult to separate and label properly. Canopies in and out of shadow cannot be separated when hand labeled on the pixel level, but grass in and out of shadow often can. Should the distinction between in and out off shadow be made if it cannot be applied between all forms of vegetation? Combining some of these classes can improve the labeling process, reduce label noise between inter/intra classes, and reduce the burden on the classifier for handling difficult class confusions.

Notable confusions are sky/ clouds, canopies/ grass/ shadows, water/ reflections on water of objects and the sky. The class selections were initially made to distinguish any object that was readily marked up. However, this study has shown significant confusion between similar classes, discussed above. The labeling process has also afforded insight into the problem, as the classes that were initially thought to be easily labeled actually provided great difficulty. In

particular, water/reflections, sky/clouds, bark/vegetation were difficult to differentiate in the labeling process. Almost all water in this study showed reflections, i.e. there was no running water, and was delineated by whether the reflections showed a discernible object. This was primarily done as the reflections of vegetation were vastly different than the reflections of the sky. In practice however, how does one find precisely where the reflected object is not discernible? This is a concept not easily communicated or made consistent. There are examples were the bottom of the pond is visible and regions on the object edges that are difficult to segment. Clouds and clear blue sky was also often difficult to label properly, as the VIS band was often overexposed resulting in a white sky. One can discern that the region is not a cloud particularly in the SWIR, but ambiguities still exist. Thin clouds and/or an overcast sky for example can often result in high ambiguity. Bark was also difficult to label, primarily because branches are often thin and obscured by noise from leaves, prohibiting segmentation. The intensities of the bark between bands differed as well, where it may have been discernible in the SWIR but not the VIS for example. The thin branches also confounded the labeling of canopies, as one could not apply one large swath of impassible vegetation without introducing label noise. Segmenting such objects that are so small and spindly is exceedingly difficult and time consuming.

These problems resulted in classes that had a shortage of labeled pixels. Simplifying the class selection to combine similar classes and remove those that cannot be easily labeled will not only increase the number of samples, but improve the workflow of the system and provide a means to reduce this class confusion.

7.1.2 Data Collection and Data Set Size

Early on in both this thesis and in previous work of the Mechatronics lab the idea that a large amount of data would be required was posited. Indeed, the question of how much data is required to reach a certain performance is quintessential to the field, and it is often the assumption that performance will increase as the number of samples are increased. This thesis sought to explore this idea using the learning curve where performance was plotted against a converging data set size. However, the expected convergence curve was not produced and it was found that the imbalance problem could influence the behavior. The class imbalance problem was addressed through the C-SVM formulation, but not during the creation of the data set stage. Future work should address the class imbalance problem by closing the loop between data collection and analysis. The distribution of the samples available with respect to the class index can be viewed after each data collection period and the next data collection can be tailored to capture the appropriate targets. Larger samples for the smaller sets used by this thesis can be increased this way. Contra to this proposed format, the data used for this work was collected in an open loop fashion. The labeling system, data collection, and classifier implementation was being created concurrently, and so this feature was not implemented.

Those seeking to maximize the data set size should also consider strongly methods to make data collection easier. Namely, 7.3 Exposure Control, 7.5 Field of View (FoV) Limitations, and 7.6 Band Combinatorics.

7.1.3 Balancing & Sub-Sampling

It is reasonable to be concerned of the class imbalance problem as this is a consistent problem within the field, there was significant confusion with the majority class, and there was poor performance of the smaller classes. While the C-SVM regularization was used to combat this problem a more direct approach would be to capture additional samples and/or subsample the larger sets. Future work should strongly consider more in depth implementation of the sub sampling process, for both the training and test set.

Such a sub sampling system would also address the concerns that the data convergence was not performed correctly. As discussed in 6.1 Data Convergence, this study split the test/train set on a per frame basis but converged on a percent of each frame basis. This was done as multiple frames are required to accumulate samples for each class, and one cannot extract pixels for testing from a frame that contributed pixels for training. Section 6.1 demonstrated that a small sub set of each frame is required to reach a performance level, and that percentage is likely very small. Increasing the percentage increased training time significantly but did not increase performance. Future work should strongly consider a more advanced sample extraction system, so that the samples can be extracted in a sub-sampled and balanced fashion.

7.1.4 Cross Validation for the Entire Data Set

This study used one test set partition and one training set partition which can cause the sampling process to affect the results. Two very similar studies by Namin and Petersson[1] plus Salamati, Larlus, Csurka, and Süsstrunk[105] both used cross validation on the *entire* data set in order to "reduce the dependency of the results to the data"[1]. This is a very promising avenue to pursue as the samples were not distributed well across all frames. The extremely narrow FoV made it difficult to expose for a large number of targets, resulting in a few types per frame. In addition, many scenes were of only a few target types to begin with. The one train/test split my not have been optimal and performing cross validation on this level would be informative, especially given the peculiarity of the convergence plots. Future work should consider strongly performing cross validation to produce combined classifier performances across k test sets. The frames could be divided into k partitions, and the class balancing and the normal operations would continue in the same process.

It is possible that this could also smooth out some of the convergence plots. While the samples were randomly selected, and the sample ratios were shown to be even, it is still possible that

the selected data influenced the system heavily. The Rock/Brick class for example was of a very small sample ratio but performed the best in the precision metric. This could also be explained by lack of confounding variables such as lighting however. If this route is pursued, it may be necessary to increase the number of frames rather than the density of the labeled pixels. This is because the samples are partitioned on a frame by frame basis. One could converge on both the number of samples per frame harvested and the number of folds, but it would likely be more informative to test the former. The number of folds is often assumed in studies, and again, the rule of thumb is a 20% for the test set, or $k = 5$. Both [105] and [1] used a 5 fold cross validation of the entire data set.

7.2 Image Registration and Exposure

Roughly half the frames captured for this work were actually registered and labeled for use. The frames that were not used were missing at least one registered image, preventing the image cube from being completed. Recall that the registration implementation was tasked with registering the SWIR and VIS/NIR POLARs to either the VIS or NIR frame of the JAI-AD080GE. The the LWIR registration was out of the scope of the work. The system often would be unsuccessful and such frames were discarded during the labeling step. Qualitatively speaking, low exposure levels of the JAI-AD080GE camera (VIS & NIR) significantly inhibited registration success. The exposure of the JAI-AD080GE was often increased in order to accommodate this problem resulting in an overexposed sky. This is not necessarily a negative aspect as one often would not be interested in that region for UGV applications. In fact, the sky will have to be overexposed in order to expose the ground properly. How can the registration and exposure issue be addressed in order to increase robustness? Future work would benefit from increasing the success of registration, especially when an embedded solution is considered.

(a) Unsuccessful VIS (b) Unsuccessful NIR (c) Successful VIS (d) Successful NIR

Figure 7.2.1: Successful registration upon a significant increase in exposure. The VIS and NIR frames are paired such that the prior pair had the same exposure level and likewise for the latter pair.

(a) Unsuccessful VIS (b) Unsuccessful NIR (c) Successful VIS (d) Successful NIR

Figure 7.2.2: Successful registration upon a slight increase in exposure. The VIS and NIR frames are paired such that the prior pair had the same exposure level and likewise for the latter pair.

Figures 7.2.1 and 7.2.2 show an example of this sensitivity to exposure level. Both figures show a frame were an increase in exposure changed the scene from an unsuccessful registration to a successful one. The difference can be quite stark, but also quite subtle.

The sensitivity to exposure also made data collection difficult. It was necessary to capture many frames at multiple exposure levels on the same scene for a single data collection run. The process was iterative and lengthy. Typically a scene would be selected, the system would be turned on, each camera would be tuned manually for an exposure level, and the polarimetric camera would be calibrated by hand. A frame would be captured, the SWIR and POLAR camera would be swept over two exposure levels, the JAI camera would be set to a higher or lower exposure and the process repeated. The system would be panned over the scene and the process repeated. Data collection would generally end when the battery ran out or panning the camera would no longer expose a new viewpoint. The system would be shutdown and the data would be viewed to select the successfully registered images. A high bias for overexposed VIS/NIR images, especially in the sky was developed from the user as these were more likely to register successfully. Note that the registration code was not integrated into the data collection platform as these two systems were developed separately. Future work would benefit from integrating these two systems.

The registration implementation also had issues with disparity. The projective transformation was used rather than the affine as there were near objects in the scene, and despite this, there still were objects that were misaligned. The projective transformation is appropriate for distances that are not very large compared to the focal length but still requires scenes that are relatively flat. This can be seen in figure 7.2.3 where a light pole near the cameras exhibits disparity and occlusion. As each pixel was assumed to be registered in the viewpoint of the classifier stage this can cause errors in labeling and feature production.

Errors in registration and exposure have been inhibitory to data collection, a source of feature and label noise, and will likely be inhibitory for application in the field as it is not robust to changes in illumination. Future work should consider strongly improvements in said registration and exposure control/metrics with the goal of addressing these issues. Mutual

(a)

(b)

Figure 7.2.3: Example of registration error with a near object. The large vertical light pole illustrates the horizontal disparity well when the two registered images are aligned vertically on the page.

.

information seems to be a promising avenue for image registration on multispectral data as image intensities are not compared directly, see [112].

7.3 Exposure Control

Exposure control is significant to stable registration as discussed in 7.2 but is also important for the comparison of image intensities between cameras and for the polarimetric camera.

The NDWI compared SWIR and Green light, yet these intensities were combined without reference to the exposure level of each camera. This means that errors in exposure level of one camera can severely impact the feature. This was not an issue for the NDVI as the JAI-AD080GE was set to the same exposure for each sensor. i.e. VIS and NIR. However, differences in sensor sensitivity were not evaluated. Future work should consider measuring the ambient light level, perhaps with a lux meter, and passing the image exposure value, sensor gain, ambient light level, and any other pertinent data to the classifier for evaluation as features.

The polarimetric camera was certainly singular in the data provided and was increasingly interesting as a source of information. However, the exposure calibration procedure was onerous. Each sensor had to be calibrated to a non polar scene in order to compensate for the difference in light levels passed by the beam splitters. This was necessary as the intensities were combined directly and the change in intensity due to the beam splitter was far greater than from the filter: "...the non-ideal beam splitter assembly [destroys] any intensity differences due to polarization. Thus manually setting each camera's exposure is necessary before taking data"[6]. The procedure was as follows. A non-polarizing scene is observed such as "...pointing the camera straight down at the ground", the camera exposures are tuned so that "the mean intensities are equal"[6]. This is complicated by the fact that more than one input is required to tune the exposure, namely gain and shutter speed, and the bounds of one often require a balance between the many choices. This is complicated again as changes in illumination can occur over the course of seconds, such as changes in cloud cover. This often required that each scene be tuned individually. Panning the system often caused one camera to fall outside the range of an acceptable exposure. This is understandable as the camera was a prototype and there were many design constraints on the system.

Future work should consider metrics for relative exposures to make intensity comparisons more robust, and should consider a more automatic way of controlling the ideal exposure level. Sweeps of exposures could also be used to provide variability in the data set as well as to make the system more robust to the exposure control algorithm used.

7.4 Additional Parameter Optimization

The following parameters were not optimized during this work.

- The base κ multiplier for sigma, which changes the filter kernel sizes. See 4.2, 5.1.3.
- The percentage used for the test/train split, see 5.1.2.
- The filter/band application map, see 5.1.3.
- The percent training data used for the coarse validation step, see 5.1.1.

Optimizing these additional parameters could provide insight into salient features as well as the upper limit of performance for this system. Future work should consider these options, in particular the filter/band application map. This parameter appears to have a significant influence on the result given the preliminary tests performed.

7.5 Field of View (FoV) Limitations

The polarimetric camera used in this thesis had a significantly smaller FoV due to constraints in the construction, namely the beam splitters and optics available. See figure 7.5.1. This resulted in a significant loss of potential samples. In addition, it complicated the data collection process as the rig had to be panned and re-calibrated many more times to capture a scene. Future work should consider strongly any options available to increase the FoV of this camera, and by extension any limiting FoV.

(a) VIS (b) SWIR (c) POLAR

Figure 7.5.1: Example registration illustrating the FoV limitations. Registered images are provided for SWIR and POLAR. The black borders represent the area where the FoV does not overlap.

7.6 Band Combinatorics

The question of which cameras were the most informative is useful as the answer can drive the inclusion or exclusion of hardware into a product. The cost/benefit of a camera can be explored when the change in accuracy can be paired with the price of a camera. The number of labeled samples can be influenced as well, shown in 7.5. If the FoVs of certain cameras limit the available training samples to a point that it becomes detrimental, it may be beneficial to not include such a camera. Future work should consider testing the combinatorics of the given bands. This can have complex interactions downstream however as certain features will be im/possible only when certain combinations are available. The code used for this thesis included scalability options in order to perform such a test, but was not implemented.

7.7 Expansion of Difference Indices

During the NDWI literature review it became apparent that NDWI in particular had many variations depending on the hyperspectral system in use. It was adapted to the bands available here and the explanation given in 2.1.2. This selection prompts the question: would other combinations of bands in the difference index strategy be informative? Combinatorics of the bands available could provide insight into the optimal combination. Such combinations would mimic the result of a kernel. Simply put, the normalized difference formula of a difference divided by sum can be applied to any combination.

7.8 Improving the Saturation over Value Formulation

The Saturation/ Value feature was not rescaled to uchar representation due to the unmanageable bounds, where one side approaches infinity. Instead the inputs were quickly displaced in order to make the computations viable as discussed in 4.4.3. This resulted in a lopsided feature space where one side was an asymptote and the other decayed. The learner can compensate for this lopsided space but it is not ideal for human consumption. This incidentally is very similar to early NDVI implementations, namely RVI of equation 2.9. The RVI was later improved to use a difference of the two inputs over the sum seen in equation 2.13 resulting in a bounded output $[-1, ..., 1]$. This space can be rescaled consistently for uchar representation during human consumption. Future work should consider modifying the original ratio of Saturation/ Value to the difference index for this benefit.

7.9 Context in Image Labeling

Context is very important for people to understand imagery. He, Zemel, and Carreira-Perpinan[17] have shown this with reflections of sky in water. Showing a small image patch from each of these regions without the surrounding objects demonstrate that the color can be very similar resulting in a very difficult problem. Future work should consider strongly incorporating a measure of context in the feature to alleviate such a problem. Texture layout filters[14] and Conditional Random Fields[14][105] are common techniques used to do so.

7.10 Data Collection with Respect to Time of Year

One variable not investigated here is the change in appearance of objects over the time of year. Previous work in the Mechatronics Lab has shown that training a system in the summer for shipment in the fall and testing in the winter can be problematic. There is a large variation in vegetation for example from the beginning to the end of Fall. Time constraints were a primary factor for this work, as the system was not operational for over a year. Such a capability is not necessarily required in research, but can be significant during application.

7.11 Polarization with Respect to Sun Position

One of the variables not considered in this work was the influence of the position of the sun with respect to the cameras. DoLP is influenced heavily by the incidence angle and causes significant variations in the data. Indeed there are many interactions not explored here: "[DoLP] is dependent on sky conditions, sun position, and viewing geometry, as well as on the polarization of light upwelling from within the water body[9]. Future work should consider this factor on the variability of the data set. The inclusion of a compass could allow one to include the position of the sun relative to the rig as a feature, of which could be a significant data source given these observations.

7.12 Processing Categorical Data

The authors of LibSVM[97] recommend expanding a categorical feature into different dimensions of the feature vector. They recommend expanding m categories into an m length vector, where the vector is of value 1 at index $m-1$ and zero everywhere else. "Our experience indicates that if the number of values in an attribute is not too large, this coding might be more stable than using a single number"[97].

Bibliography

[1] S. T. Namin and L. Petersson, "Classification of materials in natural scenes using multi-spectral images," *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1393–1398, Oct. 2012. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6386074

[2] S. U. Inc., "Images; The Advantages of Short Wave Infrared Imaging," 2014. [Online]. Available: http://www.sensorsinc.com/images.html

[3] D. M. Bradley, S. M. Thayer, A. Stentz, and P. Rander, "Vegetation Detection for Mobile Robot Navigation," *Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-04-12*, 2004. [Online]. Available: http://www.ri.cmu.edu/pub_files/pub4/bradley_david_2004_2/bradley_david_2004_2.pdf

[4] A. L. Rankin and L. H. Matthies, "Daytime Mud Detection for Unmanned Ground Vehicle Autonomous Navigation," CALIFORNIA INST OF TECHNOLOGY PASADENA JET PROPULSION LAB, Tech. Rep., 2008. [Online]. Available: http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADA505685

[5] L. B. Wolff, "Polarization vision: a new sensory approach to image understanding," *Image and Vision Computing*, vol. 15, no. 2, pp. 81–93, Feb. 1997. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0262885696011237

[6] M. Umansky, A. L. Wicks, K. Meehan, and D. W. Hong, "A Prototype Polarimetric Camera for Unmanned Ground Vehicles; Thesis submitted to the faculty of the VPI&SU in partial fulfillment of the requirements for the degree of Master of Science in Mechanical Engineering," Virginia Polytechnic Institute and State University, Tech. Rep., 2013.

[7] B. J. Goldman, A. L. Wicks, and K. Meehan, "Broadband World Modeling and Scene Reconstruction," Ph.D. dissertation, VPI & SU, 2013. [Online]. Available: http://vtechworks.lib.vt.edu/bitstream/handle/10919/23094/Goldman_BJ_T_2013.pdf?sequence=1

[8] V. A. Inc., "Velodyne Lidar," 2014. [Online]. Available: http://velodynelidar.com/

[9] L. H., R. Arturo L, and Matthies, "PASSIVE SENSOR EVALUATION FOR UNMANNED GROUND VEHICLE MUD DETECTON," Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, Tech. Rep., 2009.

[10] (Xenics Infrared Solutions), "Microbolometer Data Sheet," Xenics, Leuven, Belgium, Tech. Rep., 2013. [Online]. Available: http://www.xenics.com/documents/20120403_Gobi-384_Scientific_LowRes.pdf

[11] A. L. Rankin, L. H. Matthies, and P. Bellutta, "Daytime water detection based on sky reflections," *2011 IEEE International Conference on Robotics and Automation*, pp. 5329–5336, May 2011. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5980525

[12] J. Winn, A. Criminisi, and T. Minka, "Object categorization by learned universal visual dictionary," *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, pp. 1800–1807 Vol. 2, 2005. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1544935

[13] K. P. Murphy, *Machine Learning A Probabilistic Perspective*. Cambridge, Massachusetts: The MIT Press, 2012.

[14] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "TextonBoost for Image Understanding : Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture , Layout , and Context," *Int. Journal of Computer Vision (IJCV)*, 2007. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=117885

[15] V. A. Inc., "User's Manual and Programming Guide HDL-64E S2 and S2.1; Firmware Version 4.07," Velodyne LIDAR, Inc., Morgan Hill, CA 95037, Tech. Rep., 2012. [Online]. Available: http://velodynelidar.com/lidar/products/manual/63-HDL64ES2hHDL-64ES2CDHDL-64ES2UsersManual.pdf

[16] S. Y. Chen, H. Tong, and C. Cattani, "Markov Models for Image Labeling," *Mathematical Problems in Engineering*, vol. 2012, pp. 1–18, 2012. [Online]. Available: http://www.hindawi.com/journals/mpe/2012/814356/

[17] X. He, R. Zemel, and M. Carreira-Perpinan, "Multiscale conditional random fields for image labeling," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2. IEEE, 2004, pp. 695–702. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1315232

[18] S. J. D. Prince, *Computer vision : models , learning and inference*. Cambridge, MA: Cambridge University Press, 2012. [Online]. Available: http://www.computervisionmodels.com

[19] G. Carneiro and N. Vasconcelos, "Formulating Semantic Image Annotation as a Supervised Learning Problem," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2. IEEE, 2005, pp. 163–168. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1467437

[20] M. Schroder, H. Rehrauer, K. Seidel, and M. Datcu, "Interactive learning and probabilistic retrieval in remote sensing image archives," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 38, no. 5, pp. 2288–2298, 2000. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=868886

[21] G. A. Clark, S. K. Sengupta, W. D. Aimonetti, F. Roeske, and J. G. Donetti, "Multispectral Image Feature Selection for Land Mine Detection," *IEEE transactions on geoscience and remote sensing*, vol. 38, no. 1, pp. 304–311, 2000.

[22] JAI, "User ' s Manual AD-080GE; Digital 2CCD Progressive Scan Multi-Spectral Camera," p. 80, 2012.

[23] I. (Sensors Unlimited, "SU320HX-1.7RT," (Sensors Unlimited, Inc.), Tech. Rep., 2014. [Online]. Available: http://www.sensorsinc.com/downloads/SU320HX.pdf

[24] I. (Point Grey Research, "Firefly MV Technical Reference Manual (V1.5)," (Point Grey Research, Inc), Richmond, British Columbia V6W 1K7 Canada, Tech. Rep. 604, 2011.

[25] J. A. Richards and X. Jia, *Remote Sensing Digital Image Analysis*, 4th ed. Springer-Verlag Berlin Heidelberg, 2006.

[26] D. N. M. S. Short, "Remote Sensing Tutorial," 2005. [Online]. Available: http://fas.org/irp/imint/docs/rst/Front/tofc.html

[27] (NASA/JPL-Caltech), "AVIRIS Airborne Visible/ Infrared Imaging Spectrometer," 2014. [Online]. Available: http://aviris.jpl.nasa.gov/

[28] N. W. Service, "National Weather Service Remote Sensing Introduction," 2014. [Online]. Available: http://www.srh.noaa.gov/jetstream/remote/remote_intro.htm

[29] C. R. Nave, "Classification of Polarization," 2012. [Online]. Available: http://hyperphysics.phy-astr.gsu.edu/hbase/hframe.html

[30] I. Wikimedia Foundation, "Brewster's Angle," 2015. [Online]. Available: http://en.wikipedia.org/wiki/Brewster%27s_angle

[31] D. Barber, *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012. [Online]. Available: http://www0.cs.ucl.ac.uk/staff/D.Barber/brml/

[32] I. Netflix, "Netflix Prize," 2009. [Online]. Available: http://www.netflixprize.com/

[33] Y. Abu-Mostafa, "Learning From Data; Machine Learning Course - Recorded at a live broadcast from Caltech," 2014. [Online]. Available: http://work.caltech.edu/telecourse.html

[34] Y. Absramson, Yotam; Freund, "Active learning for visual object detection," UCSD, Tech. Rep., 2004. [Online]. Available: http://cseweb.ucsd.edu/~yfreund/papers/sevillePaper.pdf

[35] R. Garnett, J. Schneider, and R. Mann, "Bayesian Optimal Active Search and Surveying," *Proceedings of the 29th International Conference on Machine Learning (ICML 2012)*, 2012. [Online]. Available: http://arxiv.org/abs/1206.6406

[36] S. D. J. Langford;, "Active Learning Tutorial," in *Active Learning Tutorial, ICML 2009*, 2009. [Online]. Available: http://hunch.net/~active_learning/

[37] D. E. King, "Dlib-ml: A Machine Learning Toolkit," *Journal of Machine Learning Research*, vol. 10, pp. 1755–1758, 2009.

[38] C.-c. Chang and C.-j. Lin, "LIBSVM : A Library for Support Vector Machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, p. 27, 2012.

[39] "OpenCV," 2014. [Online]. Available: http://opencv.org/about.html

[40] C. Pohl and J. L. Van Genderen, "Review article Multisensor image fusion in remote sensing: Concepts, methods and applications," *International Journal of Remote Sensing*, vol. 19, no. 5, pp. 823–854, Jan. 1998. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/014311698215748

[41] G. Piella, "A general framework for multiresolution image fusion: from pixels to regions," *Information Fusion*, vol. 4, no. 4, pp. 259–280, Dec. 2003. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1566253503000460

[42] R. Luo and M. Kay, "A tutorial on multisensor integration and fusion," *[Proceedings] IECON '90: 16th Annual Conference of IEEE Industrial Electronics Society*, pp. 707–722, 1990. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=149228

[43] L. G. Shapiro and G. C. StockMan, *Computer Vision*. Upper Saddle River, New Jersey: Prentice Hall, 2001.

[44] X. Ren and J. Malik, "Learning a classification model for segmentation," *Proceedings Ninth IEEE International Conference on Computer Vision*, vol. 1, no. c, pp. 10–17, 2003. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1238308

[45] M. Everingham, L. Van˜GooL, C. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results." [Online]. Available: http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html

[46] K. W. Pratt, *Digital Image Processing*, 4th ed. Hoboken, New Jersey: John Wiley & Sons, Inc., 2007.

[47] P. Buser and M. Imbert, *Vision*. One Rogers Street Cambridge MA 02142-1209: MIT Press, 1992.

[48] G. Hoffmann, "Gernot Hoffmann Color Models RGB / HLS / HSB Contents," University of Applied Sciences in Emden, Tech. Rep., 2013. [Online]. Available: http://www.fho-emden.de/old_hoffmann/www/hlscone03052001.pdf

[49] ——, "CIELab Color Space," University of Applied Sciences in Emden, Emden, Germany, Tech. Rep., 2013. [Online]. Available: http://www.fho-emden.de/old_hoffmann/www/cielab03022003.pdf

[50] "Technical Guides Color Models CIELAB," p. 2, 2000. [Online]. Available: http://dba.med.sc.edu/price/irf/Adobe_tg/models/cielab.html

[51] A. A. Goshtasby, *2-D and 3-D Image Registration; for Medical, Remote Sensing, and Industrial Applications*. Hoboken, New Jersey: John Wiley & Sons, Inc., 2005.

[52] R. Szeliski and U. Szeliski, Richard (Microsoft Research, "Image Alignment and Stitching: A Tutorial," *Foundations and Trends in Computer Graphics and Vision*, vol. 2, no. 1, pp. 1–104, 2006. [Online]. Available: http://www.nowpublishers.com/product.aspx?product=CGV&doi=0600000009

[53] C. Bahnsen, "Thermal-visible-depth image registration," Ph.D. dissertation, Aalborg University, 2013.

[54] M. Varma and A. Zisserman, "A Statistical Approach to Texture Classi cation from Single Images," *International Journal of Computer Vision*, 2004.

[55] R. Sensing, I. Analysis, and I. N. Plant, "REMOTE SENSING AND IMAGE ANALYSIS IN PLANT," *Annu. Rev. Phytopathol*, vol. 15, pp. 489–527, 1995.

[56] F. Baret, S. Jacquemoud, and J. F. Hanocq, "The soil line concept in remote sensing," *Remote Sensing Reviews*, vol. 7, no. 1, pp. 65–82, Feb. 1993. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/02757259309532166

[57] F. J. Garc and J. Melia, "A generalized soil-adjusted vegetation index," *Re*, vol. 82, pp. 303–310, 2002.

[58] D. M. Bradley, R. Unnikrishnan, and J. Bagnell, "Vegetation Detection for Driving in Complex Environments," in *Robotics and Automation, 2007 IEEE International Conference on*, no. April, 2007, pp. 503–508.

[59] D. E. V. S. Shrestha and E. Bartlett, "Segmentation of Plant from Background using Neural Network Approach," in *Intelligent Engineering Systems through Artificial Neural Networks: Proc. Artificial Neural Networks in Engineering (ANNIE) International Conference*, vol. 11, 2001, pp. 903–908.

[60] F. D. Smedt, I. Billauws, and T. Goedemé, "Neural Networks and Low-Cost Optical Filters for Plant Segmentation," *nternational Journal of Computer Information Systems and Industrial Management Applications (IJCISIM)*, vol. 3, pp. 804–811, 2011. [Online]. Available: http://www.mirlabs.org/ijcisim/regular_papers_2011/Paper91.pdf

[61] J. M., "Absorption (light) coefficient of water: Data sources," 2007. [Online]. Available: http://www.tpdsci.com/Tpc/AbsCfOfWaterDat.php

[62] M. Wang, "Remote sensing of the ocean contributions from ultraviolet to near-infrared using the shortwave infrared bands: simulations." *Applied optics*, vol. 46, no. 9, pp. 1535–47, Mar. 2007. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/17334446

[63] A. L. Rankin, L. H. Matthies, and A. Huertas, "DAYTIME WATER DETECTION BY FUSING MULTIPLE CUES FOR AUTONOMOUS OFF-ROAD NAVIGATION," in *Proceedings of the 24 th Army Science Conference*. Orlando, Florida: JET PROPULSION LAB PASADENA CA, 2000. [Online]. Available: http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADA432750

[64] Y. Liu, "Why NDWI threshold varies in delineating water body from multitemporal images?" *International Geoscience and Remote Sensing Symposium (IGARSS)*, pp. 4375–4378, 2012.

[65] C. Liu, B. Wu, Y. Tian, W. Xu, and J. Huang, "Crop Drought Monitoring Using Serial," *Sciences-New York*, vol. 00, no. 2001, pp. 2264–2267, 2004.

[66] S. M. Razali and A. A. Nuruddin, "Assessment of water content using remote sensing Normalized Difference Water Index: Preliminary study," in *Proceeding of the 2011 IEEE International Conference on Space Science and Communication (IconSpace)*, no. July, 2011, pp. 265–268.

[67] T. Choi, J. J. Qu, and X. Xiong, "A Thirteen-year Analysis of Drought in the Horn of Africa with MODIS NDVI and NWDI Measurements," in *Agro-Geoinformatics, 2013 Second International Converence on*, 2013, pp. 302–307.

[68] Y. Wang, F. Huang, and Y. Wei, "Water Body Extraction from LANDSAT ETM + Image Using MNDWI and K-T Transformation," in *Geoinformatics, 21st International Converence on2*, 2013, pp. 1–5.

[69] (NASA/USGS), "Surface Reflectance 8-Day L3 Global 500m MOD09A1," 2014. [Online]. Available: https://lpdaac.usgs.gov/products/modis_products_table/mod09a1

[70] P. Teillet, J. Barker, B. Markham, R. Irish, G. Fedosejevs, and J. Storey, "Radiometric cross calibration of the Landsat-7 ETM+ and Landsatt-5 TM sensors based on tandem data sets," *Remote Sensing of Environment*, vol. 78, pp. 39–54, 2001. [Online]. Available: http://digitalcommons.unl.edu/cgi/viewcontent.cgi?article=1012&context=nasapub

[71] J. J. Rouse, R. Haas, D. Deering, J. Schell, and J. Harlan, "Monitoring the vernal advancement of Retrogradation (Green Wave Effect) of Natural Vegetation," Remote Sensing Center, Texas A&M University, College Station, Texas 77843, Tech. Rep. September 1972, 1974. [Online]. Available: http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19750020419.pdf

[72] R. CRIPPEN, "Calculating the vegetation index faster," *Remote Sensing of Environment*, vol. 34, no. 1, pp. 71–73, Oct. 1990. [Online]. Available: http://www.sciencedirect.com/science/article/pii/003442579090085Z

[73] C. F. Jordan, "Derivation of leaf-area index from quality of light on the forest floor," *Ecology*, vol. 50, pp. 663–666, 1969.

[74] D. M. Woebbecke, G. E. Meyer, K. V. Bargen, and D. A. Mortensen, "Color Indices for Weed Identification Under Various Soil, Residue, and Lighting Conditions," *Transactions of the ASAE*, vol. 38, pp. 259–269, 1995.

[75] G. E. Meyer, T. Mehta, M. F. Kocher, D. A. Mortensen, and A. Samal, "Textural imaging and discriminant analysis for distinguishing weeds for spot spraying," *Transactions of the ASAE*, vol. 41, pp. 1189–1197, 1998. [Online]. Available: http://cat.inist.fr/?aModele=afficheN&cpsidt=1598792

[76] G. E. Meyer and J. a. C. Neto, "Verification of color vegetation indices for automated crop imaging applications," *Computers and Electronics in Agriculture*, vol. 63, no. 2, pp. 282–293, Oct. 2008. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0168169908001063

[77] A. J. Pérez, F. López, J. V. Benlloch, and S. Christensen, "Colour and shape analysis techniques for weed detection in cereal fields," *Computers and Electronics in Agriculture*, vol. 25, pp. 197–212, 2000.

[78] A. G. Weber, "USC-SIPI Report #315 The USC-SIPI Image Database: Version 5," Signal and Image Processing Institute University of Southern California Department of Electrical Engineering, Los Angeles, CA, Tech. Rep. October 1997, 2006. [Online]. Available: http://sipi.usc.edu/database/SIPI_Database.pdf

[79] J. Malik, S. Belongie, T. Leung, and J. Shi, "Contour and Texture Analysis for Image Segmentation," *International Journal of Computer Vision*, vol. 43, no. 1, pp. 7–27, 2001.

[80] K. I. Laws and J. Way, "Rapid Texture Identification," in *Proc. SPIE 0238, Image Processing for Missile Guidance*, vol. 0238, 1980, pp. 376–381.

[81] G. Guo, S. Z. Li, K. L. Chan, H. Pan, and N. Avenue, "Texture Image Segmentation using Reduced Gabor Filter Set and Mean Shift Clustering," in *4th Asian Conf. on Comp. Vision*, 2000.

[82] D. G. Jones and J. Malik, "Computational framework for determining stereo correspondence from a set of linear spatial filters," EECS Department, University of California, Berkeley, Tech. Rep. Ii, 1992. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/1991/5784.html

[83] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," *Cvpr 2011*, pp. 1297–1304, Jun. 2011. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5995316

[84] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "LabelMe: A Database and Web-Based Tool for Image Annotation," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 157–173, Oct. 2007. [Online]. Available: http://link.springer.com/10.1007/s11263-007-0090-8

[85] M. Everingham, L. Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) Challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, Sep. 2009. [Online]. Available: http://link.springer.com/10.1007/s11263-009-0275-4

[86] M. K. Warmuth, J. Liao, G. Rätsch, M. Mathieson, S. Putta, and C. Lemmen, "Active learning with support vector machines in the drug discovery process." *Journal of chemical information and computer sciences*, vol. 43, no. 2, pp. 667–73, 2003. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/12653536

[87] F. Inc., "The FD-1665-MS7 Seven Channel Camera," 2014. [Online]. Available: http://www.fluxdata.com/products/fd-1665-ms7

[88] M. Everingham, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Development Kit," University of Leeds, Tech. Rep., 2012. [Online]. Available: http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2012/index.html#devkit

[89] "Microsoft Research Image Understanding." [Online]. Available: http://research.microsoft.com/en-us/projects/objectclassrecognition/

[90] R. L. Figueroa, Q. Zeng-Treitler, S. Kandula, and L. H. Ngo, "Predicting sample size required for classification performance." *BMC medical informatics and decision making*, vol. 12, no. 1, p. 8, Jan. 2012. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3307431&tool=pmcentrez&rendertype=abstract

[91] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *Proceedings of the 23rd international conference on Machine learning - ICML '06*. New York, New York, USA: ACM Press, 2006, pp. 161–168. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1143844.1143865

[92] C. Aguilera, F. Barrera, F. Lumbreras, A. D. Sappa, and R. Toledo, "Multispectral image feature points." *Sensors (Basel, Switzerland)*, vol. 12, no. 9, pp. 12 661–72, Jan. 2012. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3478863&tool=pmcentrez&rendertype=abstract

[93] W. T. Freeman and E. H. Adelson, "The Design and Use of Steerable Filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 9, pp. 891–906, 1991. [Online]. Available: http://persci.mit.edu/pub_pdfs/freeman_steerable.pdf

[94] P. G. R. Inc.), "FireFlyMV CMOS Camera Data Sheet," Point Grey Research Inc., Richmond, British Columbia V6W 1K7 Canada, Tech. Rep., 2012.

[95] J. Depa, M. Jonasz, and W. Hou, "TPDSci.com: a continually updated monograph of selected topics in the optics of turbid media," in *SPIE 7678, Ocean Sensing and Monitoring II, 76780W*, W. W. Hou and R. A. Arnone, Eds. Orlando, Florida, USA: SPIE, Apr. 2010, p. 7. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=763897

[96] A. Ng, "CS229 Lecture notes: PartV, SVM," Stanford, CA 94305, pp. 1–25, 2014. [Online]. Available: http://cs229.stanford.edu/materials.html

[97] C.-w. Hsu, C.-c. Chang, and C.-j. Lin, "A Practical Guide to Support Vector Classification," Department of Computer Science, National Taiwan Univer sity, Taipei, Taiwan, Tech. Rep. 1, 2010.

[98] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, Jun. 2006. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S016786550500303X

[99] M. Sokolova, "Beyond Accuracy , F-score and ROC : a Family of Discriminant Measures for Performance Evaluation," *AI 2006: Advances in Artificial Intelligence*, vol. 4304, pp. 1015–1021, 2006.

[100] V. Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, P. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, , A. and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, and E. Cournapeau, D. and Brucher, M. and Perrot, M. and Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825—-2830, 2011.

[101] W. S. Sarle, "Neural Network FAQ," 2002. [Online]. Available: ftp://ftp.sas.com/pub/neural/FAQ.html

[102] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," *2011 International Conference on Computer Vision*, pp. 2564–2571, Nov. 2011. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6126544

[103] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF : Speeded Up Robust Features," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008. [Online]. Available: www.vision.ee.ethz.ch/~surf/eccv06.pdf

[104] C. Tomasi, "CPS 296.1 Introduction to Computer Vision; Convolution , Smoothing , and Image Derivatives," Durham, NC, pp. 1–12, 2003. [Online]. Available: https://www.cs.duke.edu/courses/spring03/cps296.1/handouts/ImageProcessing.pdf

[105] N. Salamati, D. Larlus, G. Csurka, and S. Sabine, "Semantic Image Segmentation Using Visible and Near-Infrared Channels," *Computer VisionECCV 2012*, vol. Workshops, pp. 461–471, 2012.

[106] R. N. Clark, G. A. Swayz, R. Wise, K. E. Livo, T. M. Hoefen, R. F. Kokaly, S. J. Sutley, R. N. Clark, G. A. Swayze, R. Wise, K. E. Livo, T. M. Hoefen, R. F. Kokaly, and S. J. Sutley, "USGS Digital Spectral Library splib05a; Open File Report 03-395," U.S. Geological Survey, Denver, CO 80225, Tech. Rep., 2003. [Online]. Available: http://pubs.usgs.gov/of/2003/ofr-03-395/ofr-03-395.html

[107] S. Kocco and C. Capponi, "On multi-class classification through the minimization of the confusion matrix norm," *CoRR*, vol. abs/1303.4, no. 2004, pp. 277–292, 2013. [Online]. Available: http://arxiv.org/abs/1303.4015

[108] D. Plc, "Qt Poject," 2014. [Online]. Available: http://qt-project.org/

[109] M. Everingham and J. Winn, "The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Development Kit," Pattern Analysis, Statistical Modelling and Computational Learning, Tech. Rep., 2011. [Online]. Available: http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2011/devkit_doc.pdf

[110] R. Batuwita and V. Palade, "Class Imbalance Learning Methods for Support Vector Machines," in *Imbalanced Learning: Foundations, Algorithms, and Applications*, H. H. Ma and Yunqian, Eds. Wiley, 2013, ch. 6. [Online]. Available: http://www.cs.ox.ac.uk/people/vasile.palade/papers/Class-Imbalance-SVM.pdf

[111] V. Garcia, J. S. S. R. a. Mollineda, and R. A. J. M. Sotoca, "The class imbalance problem in pattern classification and learning," *Congreso Espanol de Informatica*, pp. 284–291, 2007.

[112] F. Maes, a. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens, "Multimodality image registration by maximization of mutual information." *IEEE transactions on medical imaging*, vol. 16, no. 2, pp. 187–98, Apr. 1997. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/9101328